Avtec PTP for Windows

Programmable Telemetry Processor

User's Guide

Version: 1.49

Date Prepared: July 19, 2001

Export Controls: Avtec's products and technology are controlled for export purposes by the U.S. Government pursuant to the Arms Export Control Act and the International Traffic in Arms Regulations or the Export Administration Act and the Export Administration Regulations. It is the company's responsibility to comply with the applicable export laws and regulations before engaging in export activities, including engaging in exports, reexports, or the disclosure of technical data in the U.S. to foreign persons, a deemed export.

TM 01-013

AVTEC SYSTEMS

Avtec PTP for Windows
Programmable Telemetry Processor

User's Guide

Copyright © 2001 Avtec Systems, Inc. All rights reserved. No part of this document may be reproduced without prior written permission from Avtec Systems.

The information in this document has been fully reviewed and is believed to be entirely reliable. AVTEC reserves the right, however, to modify any products herein to improve reliability, function or design. AVTEC does not assume any liability arising out of the application or use of any product or circuit described herein. AVTEC does not convey any license under its patent rights or the rights of others.

Support

Technical Support

This manual provides engineers with information necessary to operate the Avtec Programmable Telemetry Processor for Windows Software. Technical Support is available from Avtec Systems, Inc.

Avtec Systems, Inc.
Attn: Avtec PTP for Windows Technical Support
10530 Rosehaven St., Suite 300
Fairfax, VA 22030-2840
(703) 273-2211
support@avtec.com

Customer Service

Direct non-technical questions and business-related matters to Customer Support at the above number.

REVISION HISTORY

REV.	DESCRIPTION	DATE	APPD.
-00	Baseline release.	7-23-01	7-23-01

TABLE OF CONTENTS

DOCUMENT SCOPE X
AVTEC PTP FOR WINDOWS SYSTEM X
DOCUMENT ORGANIZATION XI
RELATED DOCUMENTATION XII

CHAPTER 1 INTRODUCTION 1-1

SECTION 1 AVTEC PTP FOR WINDOWS INTRODUCTION 1-1
Telemetry and Command Front-End System 1-1
Avtec PTP for Windows Applications 1-1
Avtec PTP for Windows Design Features 1-1
SECTION 2 AVTEC PTP FOR WINDOWS OVERVIEW 1-2
Avtec PTP for Windows Architecture 1-2
Theory of Operation 1-5
Avtec PTP for Windows Applications 1-7

CHAPTER 2 INSTALLATION AND SETUP 2-1

SECTION 1 OVERVIEW 2-1
SECTION 2 HARDWARE INSTALLATION (I/O BOARDS) 2-2
SECTION 3 AVTEC PTP FOR WINDOWS SOFTWARE INSTALLATION 2-2
Software System Requirements 2-2
Installing Avtec PTP for Windows Software 2-2
Start Menu Options 2-3
SECTION 4 CONFIGURING THE AVTEC PTP FOR WINDOWS SYSTEM 2-3
Step 1) Windows Settings 2-4
Step 2) Auxiliary I/O Device Driver (AUXIO) Configuration 2-9
Step 3) Frame Synchronizer/Telemetry Simulator (FSTS) Device Driver Configuration 2-13
Step 4) EEPROM Update Utility (Rev. B MONARCH E boards only) 2-22
Step 5) Avtec PTP for Windows Software License Registration 2-23
SECTION 5 UPGRADING SOFTWARE TO THE LATEST VERSION 2-25

CHAPTER 3 OPERATION AND CONTROL 3-1

SECTION 1 INTRODUCTION 3-1
SECTION 2 OVERVIEW 3-1
Avtec PTP for Windows Configuration 3-1
Avtec PTP for Windows Control 3-2
Avtec PTP for Windows Status 3-2
Where to Start 3-2
Basic Steps to Setup and Control 3-3
SECTION 3 GETTING STARTED 3-3
Avtec PTP for Windows Server and Console Startup 3-3
Logging In/Default Passwords 3-5
Server Window Description 3-5
Console Window Description and Menu Options 3-7
SECTION 4 WORKING WITH MODULES AND DESKTOPS 3-16
Avtec PTP for Windows Desktop 3-16
Available Modules and Functional Overview of Each 3-18
Creating a Desktop 3-22
Saving and Loading a Desktop 3-25
Desktop Control Functions/System Enable and Disable 3-27
SECTION 5 SERVER HIGHLIGHTS 3-28
Automated Setup and Control (Script Files) 3-29
Creating Script Files 3-29

Securing Control Sockets and TCP Server Sockets *3-30*

CHAPTER 4 CONFIGURATION AND STATUS **4-1**

SECTION 1 MODULES OVERVIEW 4-1
Module Window *4-1*
The "SEND" Button *4-2*
SECTION 2 DETAILED MODULE DESCRIPTIONS 4-3
Module 1 *8 Bit Sorter Module* *4-3*
Module 2 *16 Bit Sorter Module* *4-8*
Module 3 *ACE SFDU Formatter Module* *4-13*
Module 4 *Ames Command Encapsulator Module* *4-19*
Module 5 *Apogee PSK Modulator Module* *4-22*
Module 6 *ATM Module* *4-25*
Module 7 *Avtec Demod Bit Sync Viterbi Module* *4-28*
Module 8 *Avtec Rate Adjust Module* *4-37*
Module 9 *Avtec Serial Input Module* *4-43*
Module 10 *Avtec Serial Output Module* *4-61*
Module 11 *AXAF SFDU Formatter Module* *4-67*
Module 12 *Aydin Bit Sync Module* *4-73*
Module 13 *Aydin Bit Sync with Viterbi Module* *4-77*
Module 14 *Aydin PSK Demodulator Module* *4-83*
Module 15 *BCH Decoder Module* *4-87*
Module 16 *BCH Encoder Module* *4-90*
Module 17 *Best Source Select Module* *4-93*
Module 18 *Bit Density Correction Module* *4-98*
Module 19 *Bit Error Rate Tester Module* *4-100*
Module 20 *CCSDS Bit Stream Processor Module* *4-105*
Module 21 *CCSDS Virtual Channel Processor Module* *4-108*
Module 22 *CCSDS Virtual Channel Processor Version 2 Module* *4-116*
Module 23 *CCSDS Virtual Channel Simulator Module* *4-125*
Module 24 *CLCW Processor Module* *4-131*
Module 25 *CMOD Module* *4-136*
Module 26 *Com Port Transceiver Module* *4-141*
Module 27 *Command Verify Module* *4-145*
Module 28 *Convolutional Encoder Module* *4-154*
Module 29 *CPU Timer Module* *4-158*
Module 30 *DCS Demod Module* *4-161*
Module 31 *DDD Formatter Module* *4-169*
Module 32 *Decommutator Module* *4-179*
Module 33 *Delay Module* *4-198*
Module 34 *DST SFDU Formatter Module* *4-205*
Module 35 *Dump to Line Printer Module* *4-210*
Module 36 *Error Inject Module* *4-213*
Module 37 *EDOS Service Header (ESH) Formatter Module* *4-221*
Module 38 *File Playback Module* *4-225*
Module 39 *File Playback Version 2 Module* *4-229*
Module 40 *File Recorder Module* *4-235*
Module 41 *File Recorder Version 2 Module* *4-241*
Module 42 *File Spooler Module* *4-247*
Module 43 *Gather Scatter Module* *4-252*
Module 44 *GDP PSK Modulator Module* *4-255*
Module 45 *In Line UnZip Module* *4-259*
Module 46 *InLine Zip Module* *4-261*
Module 47 *IPDU Formatter Module* *4-264*
Module 48 *IPDU Receiver Module* *4-269*

Module 49 *ITCU Command Module* *4-275*
Module 50 *ITCU Telemetry Module* *4-279*
Module 51 *LEO-T CDH Formatter Module* *4-283*
Module 52 *LEO-T CDH Receiver Module* *4-288*
Module 53 *LEO-T TFDH Receiver Module* *4-294*
Module 54 *LEO-T TFDH Transmitter Module* *4-299*
Module 55 *Manual Command Module* *4-304*
Module 56 *NASCOM Blocker Module* *4-309*
Module 57 *NASCOM Deblocker Module* *4-316*
Module 58 *NASCOM RTP Formatter Module* *4-321*
Module 59 *NASCOM RTP Receiver Module* *4-326*
Module 60 *National Instruments GPIB Interface Module* *4-330*
Module 61 *Network Sockets Module* *4-335*
Module 62 *Network Sockets Version 2 Module* *4-340*
Module 63 *Null Transceiver Module* *4-344*
Module 64 *Odetics GPS Time Board Module* *4-348*
Module 65 *Packet Processor Module* *4-354*
Module 66 *Packet Processor Version 2 Module* *4-359*
Module 67 *PCR-2000 Receiver/Demodulator Module* *4-366*
Module 68 *PSK Summer Module* *4-371*
Module 69 *Randomizer Module* *4-375*
Module 70 *Raw Sockets Module* *4-378*
Module 71 *Software Synchronizer Module* *4-382*
Module 72 *Stripper Module* *4-387*
Module 73 *TDM Simulator Module* *4-392*
Module 74 *Time Delay Test Module* *4-400*
Module 75 *TLM-3-29 Standard Formatted Data Unit (SFDU) Module* *4-406*
Module 76 *Veda Bit Synchronizer Module* *4-423*

APPENDIX A: **A-1**

AVTEC PTP FOR WINDOWS SERVER COMMANDS A-1

APPENDIX B: **B-1**

IPDU HEADER SUMMARY B-1

APPENDIX C: **C-1**

ACRONYMS AND ABBREVIATIONS C-1

FIGURES

Figure 1-1: Avtec PTP for Windows Hardware Architecture 1-3
Figure 1-2: Avtec PTP for Windows Client-Server Communication Link 1-4
Figure 1-3: Example Avtec PTP for Windows Front-end Processor Operational Environment 1-8
Figure 1-4: Example Avtec PTP for Windows Telemetry and Command Data Flow 1-9
Figure 2-1: User Manager 2-4
Figure 2-2: User Rights Policy 2-5
Figure 2-3: Windows NT Folder Options 2-6
Figure 2-4: Windows NT Display Properties 2-7
Figure 2-5: Windows NT System Properties 2-8
Figure 2-6: Windows NT System Resources 2-9
Figure 2-7: AUXIO_Config Main Window. 2-10
Figure 2-8: AUXIO_Config Configuration Dialog 2-11
Figure 2-9: AUXIO_Config Main Window for Avtec AT-BERT 2-12
Figure 2-10: AUXIO_Config Configuration Window for Avtec AT-BERT 2-12
Figure 2-11: AUXIO_Config Update Registry Confirmation 2-12
Figure 2-12: AUXIO_Config Release Channel Confirmation 2-13
Figure 2-13: FSTS_Config Main Window 2-14
Figure 2-14: FSTS_Config Channel Configuration Dialog for AT-HSIO2 2-15
Figure 2-15: FSTS_Config Update Registry Confirmation 2-16
Figure 2-16: FSTS_Config Channel Configuration Dialog for Timecode 2-17
Figure 2-17: AUXIO_Config for Timecode 2-18
Figure 2-18: AUXIO_Config Configuration for Timecode 2-18
Figure 2-19: AUXIO_Config for No Board 2-19
Figure 2-20: Monarch Tic Reference Module 2-19
Figure 2-21: Monarch Tic Reference Configuration 2-20
Figure 2-22: Monarch Tic Reference Status 2-20
Figure 2-23: FSTS_Config for Timecode 2-20
Figure 2-24: Avtec Serial Input Configuration 2-21
Figure 2-25: FSTS_Config Release Channel Confirmation 2-22
Figure 2-26: Avtec PTP for Windows Display before License Registration 2-23
Figure 2-27: Avtec PTP for Windows Display during License Registration 2-24
Figure 2-28: Avtec PTP for Windows Display after License Registration 2-24
Figure 3-1: Server Window 3-6
Figure 3-2: Console Window at Startup 3-7
Figure 3-3: Console Window after Connecting to a Server 3-8
Figure 3-4: "System" Pull Down Menu 3-9
Figure 3-5: "Name Resolution" Sub-menu 3-10
Figure 3-6: Connect To Window 3-10
Figure 3-7: "Remote Desktop" Pull Down Menu 3-12
Figure 3-8: Server Status 3-14
Figure 3-9: Change Alarm 3-14

Figure 3-10: "Remote Module" Pull Down Menu 3-15
Figure 3-11: "Local Console" Pull Down Menu 3-16
Figure 3-12: Add Module Dialog Window 3-23
Figure 3-13: Console Window with Modules Loaded 3-24
Figure 3-14: Edit Module Connections Window 3-24
Figure 3-15: Save Desktop Window 3-26
Figure 3-16: Load Desktop Window 3-26
Figure 3-17: Load Desktop From Network Window 3-27
Figure 3-18: Dialog Window 3-28
Figure 4-1: Example Console Module Window 4-1

TABLES

Table 1-1: I/O Interfaces Supported by Avtec PTP for Windows 1-2
Table 1-2: Avtec PTP for Windows Software 1-4
Table 2-1: Program Group Options Available through Start Menu 2-3
Table 2-2: AUXIO_Config Main Window's Parameters 2-10
Table 2-3: AUXIO_Config Available Board Types 2-10
Table 2-4: AUXIO_Config Configuration Dialog 2-11
Table 2-5: FSTS_Config Main Window's Parameters 2-14
Table 2-6: FSTS_Config Channel Configuration Dialog Parameters for AT-HSIO2 2-15
Table 2-7: FSTS_Config Channel Configuration Dialog Parameters for Timecode 2-17
Table 3-1: How to Manually Start Avtec PTP for Windows Software 3-4
Table 3-2: Server Window Status 3-6
Table 3-3: "System" Pull Down Menu Options 3-9
Table 3-4: "Remote Desktop" Pull Down Menu Options 3-12
Table 3-5: "Remote Module" Pull Down Menu Options 3-15
Table 3-6: "Local Console" Pull Down Menu Options 3-16
Table 3-7: Available Modules and Their Functions 3-18

DOCUMENT SCOPE

This manual provides the information necessary to operate the Avtec Programmable Telemetry Processor (PTP) for Windows. The information provided ranges from top-level description of system control and installation guidelines to detailed descriptions of individual status and configuration fields.

AVTEC PTP FOR WINDOWS SYSTEM

The Avtec PTP for Windows is a telemetry and command front-end system. The system has a modular design that allows the user to customize setup and choose the functionality included in each system configuration. The result is an extremely flexible and powerful system.

The Avtec PTP for Windows is a PC-based system that provides high-end capabilities while maintaining a comparatively low-cost. The Avtec PTP for Windows software and PC platform support installation of, and different combinations of, several PC telemetry board products. This flexibility further increases system capabilities and provides diverse functions. The supported hardware includes both Avtec and other vendor products.

System throughput capabilities reflect the hardware that is installed in the system. Avtec telemetry hardware products have the following rate capabilities:

- MONARCH E — 25 Mbps
- AT-HSIO2 — 2 Mbps
- Avtec Demod/Bit Sync/Viterbi — 5 Mbps
- MONARCH EX — 320 Mbps (details are not included in this document)

The broad range of functions provided by the Avtec PTP for Windows include multiple telemetry stream input; time-tagged frame or packet data output; output to a network; high-speed data logging for store and forward operations; and command uplink.

DOCUMENT ORGANIZATION

This manual is grouped into four chapters:

Chapter 1: Introduction
Introduces the user to the Avtec PTP for Windows concepts and architecture.

Chapter 2: Installation and Setup
Walks the user through software and hardware installation processes.

Chapter 3: **Operation and Control**
Defines the GUI main menu and outlines operational procedures.

Chapter 4: Configuration and Status
Provides the configuration and data processing status details.

Several Appendices are attached to provide details that are useful in understand and configuring data processing solutions:

Appendix A provides a list of all Avtec PTP for Windows Server commands.

Appendix B provides summaries for various data and command encapsulation formats.

Appendix C provides a list of acronyms and abbreviations.

This User's Manual also provides a list of related documentation—functional descriptions and other manuals—written about various components of the Avtec PTP for Windows. Acronyms and abbreviations used in this manual are defined in the Acronyms and Abbreviations list.

RELATED DOCUMENTATION

The following documents contain information you may find useful to reference as you read this manual:

1) Avtec Systems, Inc. *AV_PTP Programmable Telemetry Processor Remote Interface Library Programmer's Guide*. Fairfax, Virginia, 1997.
2) Avtec Systems, Inc. *AT-HSIO2 High Speed Serial I/O Board with Frame Synchronizer Functional Description*. Fairfax, Virginia, 1995.
3) Avtec Systems, Inc. *MONARCH-E Frame Synchronizer/Telemetry Simulator with Reed-Solomon Encoder/Decoder Functional Description*. Fairfax, Virginia, 1997.
4) Avtec Systems, Inc. *Avtec Demodulator/Bit Synchronizer/Viterbi Decoder Board Functional Description*. Fairfax, Virginia, 1999.
5) Avtec Systems, Inc. *PTP EX User Manual*. Fairfax, Virginia, 1999.
6) Avtec Systems, Inc. *Telecommand*. Fairfax, Virginia, 1999.

Additional Related Documentation

1) Apogee Labs, Inc. *Model ISA-PSK PSK Modulator User's Manual.* North Wales, Pennsylvania, 1996.
2) Apogee Labs, Inc. *Model ISA-STG2 Synchronized Time Generator User's Manual.* North Wales, Pennsylvania, 1996.
3) Aydin Corporation, *PC335 Bit Synchronizer Installation and Operation Manual*. Horsham, Pennsylvania, 1996.
4) Aydin Corporation, *PC329 Tunable BPSK Demodulator Installation and Operation Manual*. Horsham, Pennsylvania, 1996.
5) GDP Space Systems, *PSK004 ISA-BUS BPSK Modulator User's Manual*. Horsham, Pennsylvania, 1995.
6) Microdyne Corporation. *Model PCR-2000 Telemetry PC Receiver Operations Manual*. Ocala, Florida, January 1996.
7) Veda Systems, Inc. *DB200 Convolutional Decoder Module Technical Reference Manual.* California, Maryland, September 1994.
8) Veda Systems, Inc. *Series-2/10 and DBS200 PC Bit Synchronizer Card Technical Reference Manual*. California, Maryland, January 1996.

Chapter 1
Introduction

SECTION 1
AVTEC PTP FOR WINDOWS INTRODUCTION

TELEMETRY AND COMMAND FRONT-END SYSTEM

AVTEC's Programmable Telemetry Processor (PTP) for Windows is a PC-based, telemetry and command front-end processor. It is a multi-channel system. The Avtec PTP for Windows supports both Time-division Multiplexed (TDM) and Consultative Committee for Space Data Systems (CCSDS) packetized telemetry formats, which results in a single front-end system solution that supports multiple spacecraft.

AVTEC PTP FOR WINDOWS APPLICATIONS

The Avtec PTP for Windows is an ideal solution for an application that requires a front-end system that performs data acquisition, real-time network transfer, and store and forward operations. In a typical telemetry application, the Avtec PTP for Windows acts as a gateway that accepts multiple telemetry streams and outputs time-tagged frame or packet data over a network to workstations in a distributed satellite control and analysis system. The Avtec PTP for Windows also includes a command gateway that accepts input from the network and outputs serial commands to the uplink.

The Avtec PTP for Windows is useful throughout the lifecycle of a satellite system: it provides vital functions from development to deployment. During development, integration, and test, the front-end system acts as a test tool in a distributed test environment. During operations, the system is installed at remote ground stations, with network links back to operations center(s) for telemetry and command processing and analysis.

AVTEC PTP FOR WINDOWS DESIGN FEATURES

The Avtec PTP for Windows has a modular design that provides flexibility and a powerful range of capabilities. The operator can choose the functionality that is included in each Avtec PTP for Windows application, and then easily change from one configuration to another to support numerous missions or different requirements. The PC-based design allows support of numerous PC telemetry boards, which are both Avtec and other vendor products. A key element of most

Avtec PTP for Windows applications is the AVTEC MONARCH-E card, which is a PCI-based CCSDS/TDM Telemetry Processor/Simulator board.

The system is controlled and monitored with either the "Console" Graphical User Interface (GUI) that is part of the Avtec PTP for Windows software, or via Remote Interface Library (RIL) calls. The Console GUI can run either locally or from a remote PC via a TCP/IP socket interface. The RIL provides for easy TCP/IP remote control and allows the user to integrate a custom Avtec PTP for Windows interface into an existing control environment.

Many hardware combinations and options exist for possible Avtec PTP for Windows system configurations. The Avtec PTP for Windows software capabilities are the same within each version. However, the hardware functions, which include I/O capabilities and define the specific number of channels, depend on the hardware chosen and installed to support the Avtec PTP for Windows software.

SECTION 2
AVTEC PTP FOR WINDOWS OVERVIEW

AVTEC PTP FOR WINDOWS ARCHITECTURE

The Avtec PTP for Windows is a PC-based system running under the Windows NT operating system. The PC contains PCI and ISA Input/Output (I/O) busses, includes the standard PC peripherals, and allows installation of application-specific interface boards. The number of available ISA and PCI slots in the PC determines the maximum number of I/O boards that can be installed in the system. The Avtec PTP for Windows software controls and monitors all aspects of the PC configuration.

Table 1-1 shows the standard and application-specific interfaces supported by the Avtec PTP for Windows.

Table 1-1: I/O Interfaces Supported by Avtec PTP for Windows

Application-specific Interfaces	Vendor/product name
Frame Synchronizer/Telemetry Simulator	*Avtec AT-HSIO2,* *Avtec MONARCH*
Frame Synchronizer/Telemetry Simulator with Reed-Solomon Encoder/Decoder	*Avtec MONARCH-E,* *Avtec MONARCH-EX*
Bit Synchronizer	*Avtec Bit Sync*
Demod Bit Synchronizer with Viterbi decoder	*Avtec Demod Bit Sync with Viterbi*
Command BPSK Modulator	*Apogee ISA-PSK*
Command Modulator	*ATSC Command Modulator*
PSK Summer	*ATSC PSK Summer Card*

Receiver/Demod	*Microdyne PCR 2000*
Tunable BPSK Modulator	*General Data Products PSK004*
Tunable Subcarrier BPSK Demodulator	*Aydin PC329*
IRIG-B and NASA-36 Time Code Processor	*Apogee ISA-STG2*
ATM OC-3	*Forerunner LE ATM Adapter OC-3*
Standard PC Interfaces	
Ethernet interface	
SCSI and IDE hard disks	
CD ROM drive	
COM Port	
LPT Port	

Avtec PTP for Windows Hardware

The hardware architecture is shown in **Figure 1-1**. The figure shows an example of the application interfaces that can be included in an Avtec PTP for Windows. The proposed application determines which of the application-specific interfaces are required and installed in the system.

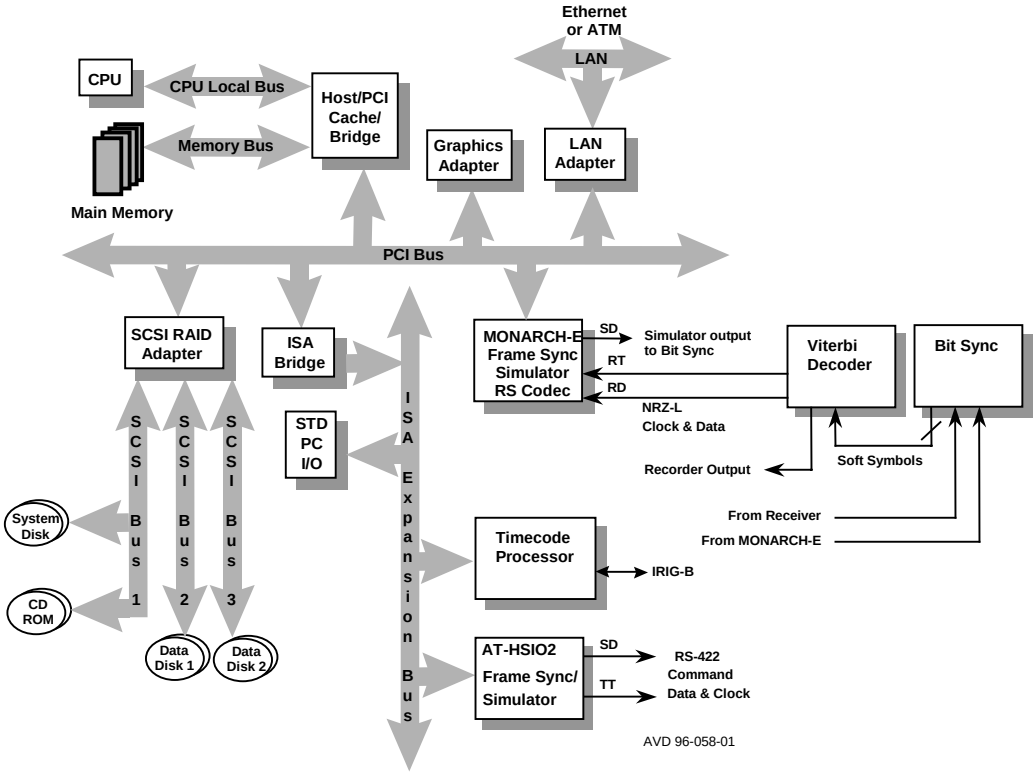


Figure 1-1: Avtec PTP for Windows Hardware Architecture

Avtec PTP for Windows Software

Avtec PTP for Windows Software Overview

The Avtec PTP for Windows software consists of a server application (PTP NT.EXE), a Graphical User Interface (CONSOLE.EXE), and the module dynamic link libraries (DLL). The Server controls the hardware and performs all of the real-time data processing. The Console application provides an easy-to-use graphical interface for controlling the configuration and monitoring status of the Avtec PTP for Windows system.

The Console communicates with the Server through a TCP/IP socket connection. The Console can run locally on the Avtec PTP for Windows system or on a remote PC that is connected to the system via a TCP/IP network. The Avtec PTP for Windows application can also be remotely controlled by a user application through the RIL.

Table 1-2 defines the three main components of the Avtec PTP for Windows software: Server, Console, and DLL's.

Table 1-2: Avtec PTP for Windows Software

Name	Summary	Description
PTP NT.EXE	Server application	The Server controls the Avtec PTP for Windows hardware and performs all of the real-time data processing. The Avtec PTP for Windows application can be controlled locally or remotely. It is controlled by either the Console or a custom, user application.
CONSOLE.EXE	GUI	The Console application provides an easy-to-use interface for controlling the configuration of the Avtec PTP for Windows system. The Console communicates with the Avtec PTP for Windows through TCP/IP sockets. It can run locally on the Avtec PTP for Windows system or on a remote PC that is connected to the system via a TCP/IP network
DLL's	modules	Dynamic Link Libraries

Server and Console Communication

Figure 1-2 illustrates the communication link between the Console *(client)* and the Server (the Server name is PTP_NT). The Server supports a single point of control, which means that only one client can control the Server at a time. However, multiple clients can simultaneously receive status from a Server to monitor its operation.

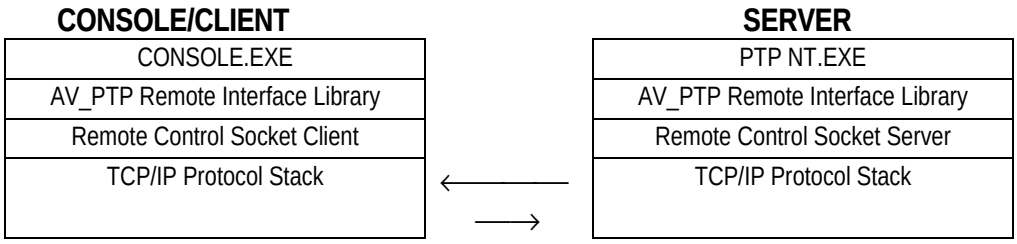


Figure 1-2: Avtec PTP for Windows Client-Server Communication Link

To remotely access the Avtec PTP for Windows system, a client (Console) must know the IP address and socket number of the targeted Server (PTP_NT). Whenever it receives a remote command, the Avtec PTP for Windows responds with a message, even if only to report an error or give status. The remote Avtec PTP for Windows does not require the destination address because it acts as a Server. It allows connections instead of initiating them *(during other non-Remote Control communications, the Avtec PTP for Windows may act as either a client or a Server).*

The Avtec PTP for Windows status messages can be configured as multicast (multicast is enable or disabled by configuration; the default is disable). If status messages are configured for multicast, the server allows multiple control devices to monitor at the same time. Console clients detect these broadcast messages to determine how many Avtec PTP for Windows systems are on the network. In addition, users can restrict access to Avtec PTP for Windows control ports and TCP Server sockets ports by creating *Well-Known Hosts* files. Please see **Chapter 3, Section 5** for more information on creating these security files.

The Console client is not required to operate the Avtec PTP for Windows system. A user may develop custom client applications to control the Avtec PTP for Windows system using the AV_PTP RIL. The RIL is available for Windows NT and Solaris. For more information about AV-PTP, please refer to Avtec's AV-PTP Programmer's Guide.

THEORY OF OPERATION

Data Processing

The Avtec PTP for Windows software modules provide numerous data processing functions along with controlling Avtec and third party hardware. Some modules also provide for data encapsulation. (**Chapter 4** provides the details of each available software module). Many of the fundamental data processing functions are provided by the custom Avtec hardware that can be installed in the PC. For most telemetry applications, the Avtec PTP for Windows acquires telemetry data streams using the Avtec Demodulator/Bit Synchronizer/Viterbi Decoder (DBV) Card and either the AT-HSIO2 or MONARCH-E (MONARCH) card. Both the AT-HSIO2 and the MONARCH-E perform frame synchronization using an adaptive strategy, serial-to-parallel conversion, and time tagging.

The AT-HSIO2, MONARCH, and MONARCH-E are bi-directional boards that can also support Telemetry Simulation, Command Output, and Bit Error Rate Testing (BERT). Both MONARCH and AT-HSIO2 boards are more like network interface cards than traditional frame synchronizer/decom cards in that they provide a stream of frame data to the PC rather than individual raw measurements and tags.

MONARCH and MONARCH E

The MONARCH-E supports rates up to 25 Mbps with hardware support for derandomization, frame synchronization, CRC, and Reed-Solomon decoding. The MONARCH-E Telemetry Simulator performs Reed-Solomon encoding, interleaving, CRC encoding, sync marker insertion, pseudo-randomization, and convolutional encoding at data rates up to 25 Mbps.

AT-HSIO2

The AT-HSIO2 accepts and outputs data at rates up to 2 Mbps. The boards also support bit error rate testing and data quality monitoring for testing and pre-pass verification functions.

Avtec DBV

The Avtec Demod Bitsync Viterbi board is a high-performance Demodulator, Bit Synchronizer and Viterbi Decoder for the PCI bus. The DBV accepts RS-422 or TTL data and outputs the recovered data and clock signal. The Viterbi Decoder supports the CCSDS and INTELSAT standard Rate ½, Constraint Length 7 convolutional code. It accepts 3-bit soft symbols at rates up to 25 Msymbols per second and outputs recovered bit data and clock. The Viterbi Decoder provides built-in Synchronization and Bit Error Rate monitoring circuits.

System Control and Monitoring

Two interfaces provide setup, control, and status information to the user: AV_PTP RIL and Console software. The AV_PTP RIL is thoroughly documented in the *AV_PTP Remote Interface Library Programmer's Guide* and only minimal information is provided about it in this user's manual. Instead, focus is on the Console software that provides a GUI to the Server, which in turn provides complete control of the Avtec PTP for Windows system.

Console and Server

Systems are delivered with the Avtec PTP for Windows Console and Server available through the PC Start/Programs menu. Since the Console and Server do not have to be running on the same PC, the Console requires the operator to connect to the Server by the IP address and port number of the PC on which the Server is running.

The Console provides the user with enable (start processing), disable (stop processing), configuration, and status tools. The Server, although not a GUI, runs with a window open that also provides low-level software and status information.

Desktop

Using the Console, the user designs desktops, which are the configuration files that are saved and contain the information to setup Avtec PTP for Windows components. Desktops also provide access to the status information once system processing begins.

The user either loads an existing desktop, creates a new desktop, or modifies an existing desktop. The desktop contains the functions and configuration parameters chosen by the operator to control the Server. The foundation of the desktops is dll's called modules.

Modules and Connections

Each module either controls an application-specific interface board or provides a specific software function. A module is added to a desktop from a selection list that provides all Avtec PTP for Windows modules, and once it is chosen shows in the Console as a Module Window.

The user chooses which modules to add to a desktop depending on the available application-specific boards and the required processing. Each module has an input and an output to it. Unless the I/O is an external interface (connector), a file, or a socket, the user must connect each module's I/O ports to another module, which directs the data flow. The number of I/O ports available is module dependent. The Console GUI provides a connection table that maps each module by an assigned number and provides switches that connect I/O together.

Config, Status, and Command Windows

Each Module Window provides standard buttons that access configuration, status, and command windows. From the Module Window Configuration button, the user has access to all configuration switches and setup parameters. The Status window provides the user real-time data processing results once data flow through the system begins. The Command window, which is only available for a few modules, provides real-time processing control.

FSTS Driver

A key component of the Avtec PTP for Windows software is the driver that controls the Avtec application-specific boards: AT-HSIO2, MONARCH, and MONARCH E. This is the FSTS Driver. The Server software interfaces with the Driver. The Server/Driver interface is invisible to the user.

AUXIO Driver

The AUXIO Driver controls the non-Avtec boards. The Server software interfaces with the Driver. The Server/Driver interface is invisible to the user.

AVTEC PTP FOR WINDOWS APPLICATIONS

This section provides two example applications for the Avtec PTP for Windows system. One example illustrates the Avtec PTP for Windows system as a vital component of an operational environment. The second example shows the Avtec PTP for Windows system as an integral component of satellite test and verification.

Operational Front-end Telemetry and Command Processor

Ground Network Application

The Avtec PTP for Windows acts as a gateway between a network and the spacecraft uplink/downlink.

Figure 1-3 shows the Avtec PTP for Windows in a typical operational environment where multiple remote ground stations are connected to control centers via a wide area network (WAN).

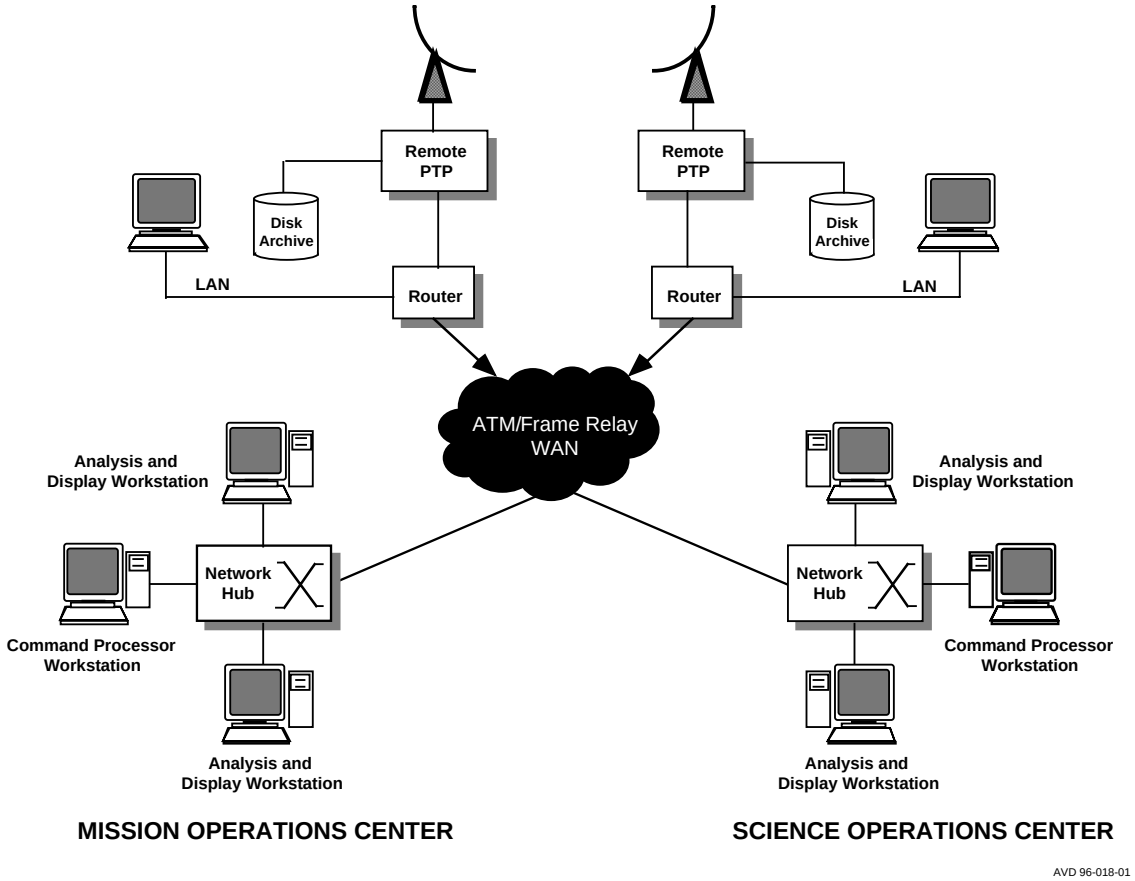


Figure 1-3: Example Avtec PTP for Windows Front-end Processor Operational Environment

Details of Avtec PTP for Windows Front-end Telemetry and Command Processing

The Avtec PTP for Windows can accept serial telemetry stream inputs, and then route them in various ways, which is especially useful for front-end telemetry and command systems. **Figure 1-4** shows the data flow and Avtec PTP for Windows modules used in an example front-end Telemetry and Command Processor application.

The figure illustrates a scenario that includes three application-specific interfaces: a Bit Synchronizer/Viterbi Decoder, an Avtec MONARCH E, and an AT-HSIO2. Both a telemetry data path and control data path is shown in the figure. The modules used (each module is assigned a number MO1 to MO9) for the telemetry data flow are Bit Synchronizer/Viterbi Decoder; Serial Input; Virtual Channel Processor; File Recorder; IPDU Formatter; and Socket. The command path uses the IPDU Receiver and Serial Output Modules.

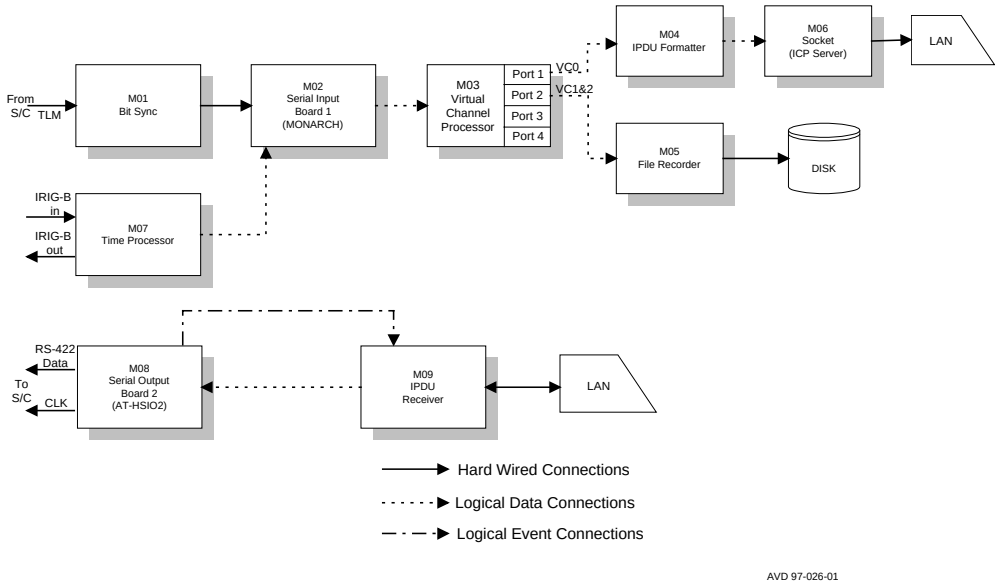


Figure 1-4: Example Avtec PTP for Windows Telemetry and Command Data Flow

The following outline describes the telemetry data path illustrated in **Figure 1-4:**

1) Avtec PTP for Windows accepts a serial telemetry stream using the Bit Synchronizer with Viterbi Decoder (BVD), which is hardware supported by an Avtec PTP for Windows software module. Clock and data output from the BSVD is input via cabling to the MONARCH-E, which is also a piece of hardware supported by an Avtec PTP for Windows software module.

2) The Serial Input Module controls the operation of the MONARCH-E input functions. The MONARCH-E performs frame synchronization, de-randomization, Reed-Solomon decoding, and time tagging. The Serial Input Module outputs telemetry frames with time tag and quality annotation to the CCSDS VCP Module.

3) The CCSDS VCP Module filters frames based on VCID and data quality. The CCSDS VCP Module has multiple data output ports. Each data output port can be configured to output a particular subset of Virtual Channels. In this example, the VCP port 1 is configured to pass only frames with VCID 0 and is connected to the IPDU Formatter Module. The VCP port 2 is configured to pass VCID 1 or 2 frames to the Recorder Module.

4) The Internet Protocol Data Unit (IPDU) Formatter receives Virtual Channel Data Units (VCDU) from the VCP port 1 and prepends an IPDU header to the data. The IPDU Formatter is connected to a Socket Module that is configured as a TCP server.

5) The Socket Module sends VCID 0 frames with IPDU header over the network in real time.

6) The VCID 1 and 2 frames output from VCP port 2 are sent to the Recorder module, which stores the data to disk.

For the command data flow, the Avtec PTP for Windows accepts commands from the network using an IPDU Receiver Module. The IPDU Receiver Module is a special type of Socket module that supports IPDU encapsulated data. It is configured as a TCP server and accepts IPDU encapsulated Command Link Transmission Units (CLTU) from a network client. The IPDU receiver strips the IPDU header from accepted network packets and passes the command data to the Serial Output Module. The Serial Output Module serializes the data for uplink using the AT-HSIO2 (Avtec hardware) serial output.

Spacecraft Test and End-to-End Test Functions

The Avtec PTP for Windows provides a full suite of tools for satellite and ground system development, integration, and testing. The extensive set of Avtec PTP for Windows test tools offer functions that are required throughout the different life cycle stages of satellite test and deployment. As a result, different life cycle stages do not require integration of different equipment, which reduces cost and eliminates compatibility issues. In addition, using the Avtec PTP for Windows in a distributed test environment enables early integration and remote access to live test data.

The Avtec PTP for Windows provides the following test functions:

- Bit Error Rate Testing (BERT): As a BERT, the Avtec PTP for Windows transmits and receives (verifies) a pseudo-random pattern (2047), which is useful for end-to-end system testing.
- Telemetry Simulation (TDM and CCSDS): Both TDM and CCSDS data can be generated and used for system self-test or output for verification of other equipment. Telemetry Simulation is critical in testing ground station systems during development and integration.
- Data Quality Monitoring (TDM and CCSDS): A full suite of statistics are generated and maintained real time. Data quality monitoring (DQM) functions can verify communications links during development and integration, and are essential for maintaining statistics on operational links.
- Data Logging and Playback: The Avtec PTP for Windows logs telemetry data at high rates as it is collected. The Avtec PTP for Windows will playback the logged data for more realistic ground system testing or for regression testing during development.
- Network Protocol Conversion: The Avtec PTP for Windows provides bi-directional network connections. This means that data can be provided to another machine on the network and it can be received from a network connection for verification. For example, a test flow can log data while gathering and viewing DQM statistics in real time, and sending filtered data over the network to a remote processing station. After the test, the filtered data can be accessed via the network for further analysis and processing.

Chapter 2
Installation and Setup

SECTION 1
OVERVIEW

This section details Avtec PTP for Windows software installation and briefly overviews hardware installation. For systems that are delivered setup and configured, most of the information in this section is not necessary to follow before beginning system use. However, it is still useful reference material.

A configured system already has the required application-specific hardware installed, and the Avtec PTP for Windows software can be accessed through the start menu. For these systems, the operator can proceed to **Chapters 3 and 4** to access the details of system operation.

The following list provides a brief overview of Avtec PTP for Windows installation and setup:

1) Install Hardware: Power-down the PC and install the application specific I/O boards such as other-vendor Aux I/O boards and Avtec I/O boards. Be sure to write down the I/O base address settings for each of the ISA boards. *NOTE*: The AT-HSIO2 boards should be configured with an I/O address that is on a 32-byte boundary within the range 0x200 - 0x3FF.
2) Log On as Administrator: Power-up the PC and log on as a user with administrator privileges.
3) Install the Avtec PTP for Windows Software: To load, run SETUP.EXE from the installation disk 1. SETUP.EXE copies all of the Avtec PTP for Windows files to the computer. *NOTE*: For software updates, unless the previously installed Avtec PTP for Windows software is in the correct directory (\PTP_NT directory) it is not overwritten and can result in confusion.
4) Configure System: System configuration includes several operating system and specific configuration steps. The two key Avtec PTP for Windows configuration steps follow:
   - *Update Registry for Aux I/O Boards*: To update the registry for the (other-vendor) Aux I/O boards, run AUXIO_Config.exe. Set the BoardType, IOAddress, and NumPorts for each channel.

- *Update Registry for Avtec I/O Boards:* To update the registry for the Avtec I/O boards, run FSTS_Config.exe. For the AT-HSIO2 boards, set the BoardType, IOAddress, Irq, RxDrq and TxDrq (DMA chan). For the MONARCH-E boards, set the BoardType.

5) Reboot and Verify Changes: Reboot the computer. Check the Windows event log to verify that the Aux IO and FSTS drivers recognize all of the devices. If there are any error messages in the event log, verify that the base address switch settings agree with the I/O Address entries in Auxio_Config and Fsts_Config.

SECTION 2
HARDWARE INSTALLATION (I/O BOARDS)

The Avtec PTP for Windows system is usually shipped with all the boards already installed. You should not have to modify any hardware inside the system. Any connections are done externally. If you order the software to install on your own system, you should follow the third party's hardware installation instructions on how to install and set up the hardware boards.

SECTION 3
AVTEC PTP FOR WINDOWS SOFTWARE INSTALLATION

This chapter describes how to install and configure the Avtec PTP for Windows software. If you have received a fully configured system, you may skip to the **Chapter 3: Operation and Control**.

SOFTWARE SYSTEM REQUIREMENTS

The Avtec PTP for Windows software is a collection of application programs and dynamic link libraries that are designed to install on a PC running the Windows NT system. The minimum system requirements for the Avtec PTP for Windows software are a PC with a Pentium 133 MHz processor and 32 MB of RAM running Windows NT 4. For data rates in excess of 5 Mbps, a Pentium II processor with at least 64 MB of RAM is recommended.

INSTALLING AVTEC PTP FOR WINDOWS SOFTWARE

The Avtec PTP for Windows software is distributed on a CD-ROM.

To install the Avtec PTP for Windows software, perform the following steps.

1) Logon as a user with administrator privileges.
2) Close all PTP software. Installation can not complete if any PTP software is running (be certain to close the Monitor program, which is a PTP application that is often running and overlooked).
3) Run SETUP.EXE from the installation disk 1. The Setup program will copy all of the Avtec PTP for Windows files to your computer.

START MENU OPTIONS

Avtec PTP for Windows software installation creates a PTP NT program group in the Start Menu under Programs. The options available via Start → Programs → PTP NT path appear in **Table 2-1**.

Table 2-1: Program Group Options Available through Start Menu

Item	Description
AUXIO_Config	AUXIO (Auxiliary I/O) device driver Registry configuration utility.
Console	Avtec PTP for Windows Console graphical user interface.
Eeprom_Update	Command line program used to update the MONARCH-E onboard EEPROM from version 1.0 to 1.1. Instructions for running the Eeprom_update utility are given below. This update only applies to Rev. B MONARCH boards.
FSTS_Config	FSTS (Frame Synchronizer/Telemetry Simulator) device driver Registry configuration utility.
FSTSLoad	Checks to see if the FSTS driver loads properly and loads the driver if it is not. This application is rarely used.
ListServers	Utility that displays the IP addresses of Avtec PTP for Windows Servers and associated clients that are detected on the local network.
Monitor	System remote control and monitor application. For more details, please see the AV_PTP Remote Interface Library Programmer's Guide.
MyConsole4000	Connects to port 4000 directly.
MyConsole5000	Connects to port 5000 directly.
PTP_NT	Avtec PTP for Windows Server window.
Readme	Readme file with release notes.

SECTION 4
CONFIGURING THE AVTEC PTP FOR WINDOWS SYSTEM

The Avtec PTP for Windows system must be configured before operating the software. To configure the system, follow steps one through five:

1) Setup Miscellaneous Windows settings

2) Configure the Auxiliary I/O Device Driver (AUXIO)
3) Configure the Frame Synchronizer/Telemetry Simulator (FSTS) Device Driver
4) Run the Eeprom_Update Utility (only for Rev. B MONARCH boards)
5) Register the software: run Avtec PTP for Windows Software License Registration

If no application specific hardware (Avtec AT-HSIO2 or MONARCH boards or Auxiliary I/O devices) is installed in the system, then steps 2 through 4 may be omitted.

STEP 1) WINDOWS SETTINGS

The Avtec PTP for Windows software requires that the Avtec PTP for Windows User Privilege and several general setup options be correctly set. The software also requires that the TCP/IP protocol and network adapter be installed. Optionally, Windows can be set up to provide automatic user logon, which brings an operational system up with the required user at boot.

Avtec PTP for Windows User Privileges

A user must have Power User or Administrator privileges to run Avtec PTP for Windows. At a minimum, it must have the "Increase scheduling priority" right. This right is granted by default to the Administrators group, so if the Avtec PTP for Windows User belongs to this group, it will automatically inherit this right.

Windows NT:

1) Go to Start → Programs → Administrative Tools → User Manager.
2) From the menu, select Policies → User Rights.

Figure 2-1: User Manager

3) Make sure the box "Show Advanced User Rights" is checked. From the Right drop down list, select "Increase scheduling priority". This right should be granted to the "Administrators" and "Power Users" group. The Avtec PTP for Windows User should belong to these groups.

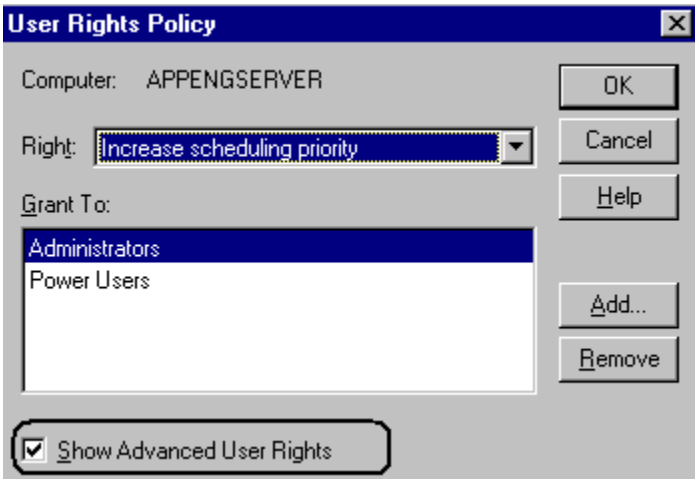


Figure 2-2: User Rights Policy

General Windows Setup Options

The Avtec PTP for Windows software also requires that several Windows options be configured as outlined below:

Windows Explorer for Windows NT:

1) Select View → Options.
2) Select the View tab. Select "Show all files."

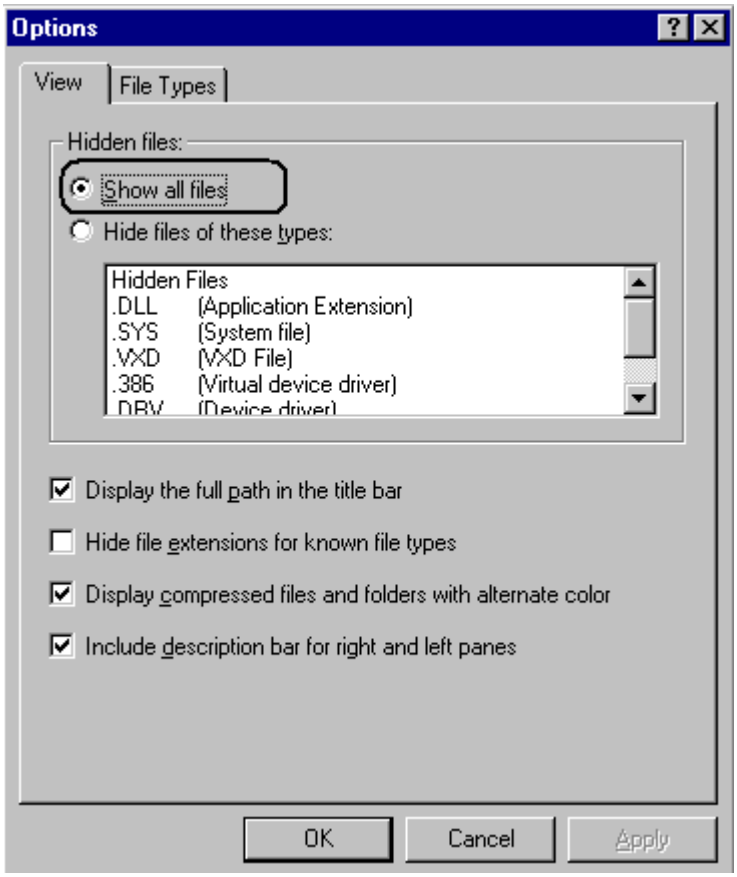


Figure 2-3: Windows NT Folder Options

Display Properties for Windows NT:

1) Open up the Control Panel. Open or double click on the Display folder.
2) Select the Plus! tab. Uncheck the "Show window contents while dragging" and "Smooth edges of screen fonts" options. It is recommended to leave these boxes unchecked to minimize CPU usage.

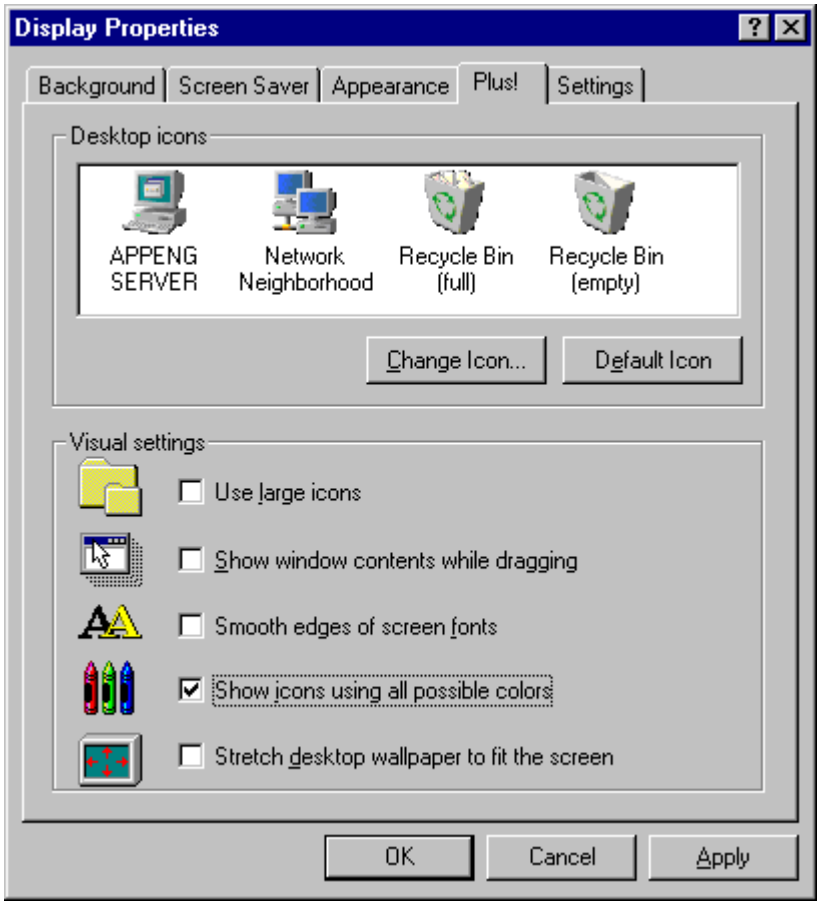


Figure 2-4: Windows NT Display Properties

System Properties for Windows NT:

1) Open up the Control Panel. Open or double click on the System folder.
2) Select the Performance tab. Set "Application Performance Boost" to "None". This option is recommended to minimize CPU usage.

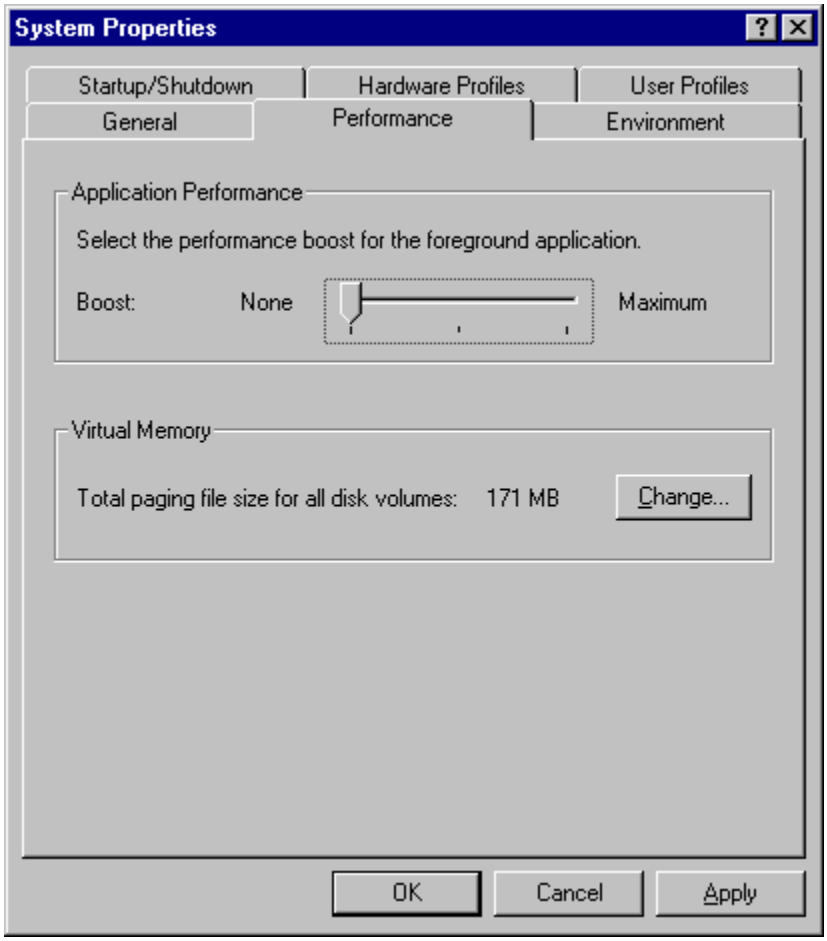


Figure 2-5: Windows NT System Properties

TCP/IP Installation

The Avtec PTP for Windows software requires that the TCP/IP protocol be installed on the computer. If no network adapter is installed in the computer, the MS Loopback Adapter can be used in conjunction with the TCP/IP protocol. For more information, please refer to the Windows NT User's Manual.

Windows System Resources

The Windows Administrative Tools provide useful information about the allocation of system resources (I/O address, IRQ lines, and DMA channels). These settings and information can be used later when running the AUXIO_Config and FSTS_Config programs.

Windows NT:

1) Go to Start → Programs → Administrative Tools → Windows NT Diagnostics.
2) Go to the Resources tab, where you will find all your system resources.

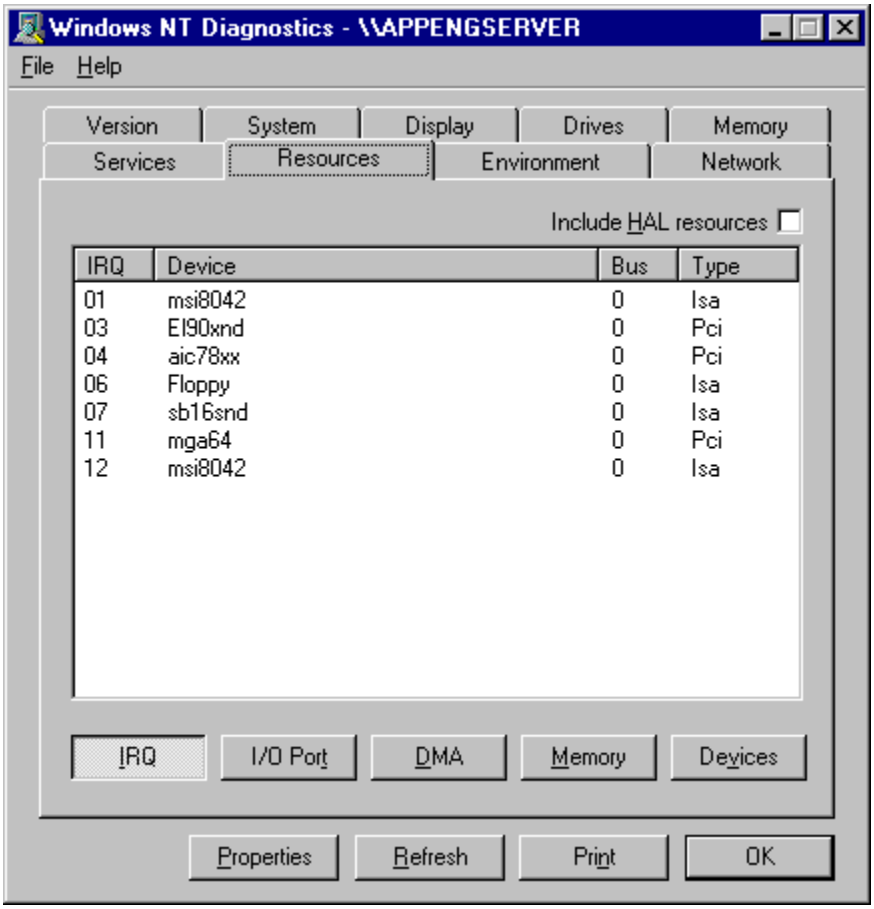


Figure 2-6: Windows NT System Resources

STEP 2) AUXILIARY I/O DEVICE DRIVER (AUXIO) CONFIGURATION

If any auxiliary I/O boards are installed in the Avtec PTP for Windows system, then these devices MUST be configured using the AUXIO_Config program. If the system does not contain any auxiliary I/O devices, then this section may be skipped. Auxiliary I/O boards that need to be configured with the AUXIO_Config program include the following:

- ✓ Avtec Demod/Bit Sync
- ✓ Veda Viterbi Decoder
- ✓ Aydin Bit Sync
- ✓ GDP PSK Modulator
- ✓ Aydin Demodulator
- ✓ Apogee PSK Modulator
- ✓ Avtec AT-BERT
- ✓ Microdyne PCR 2000
- ✓ Veda Bit Sync

The Avtec PTP for Windows software installation includes the AUXIO_Config program, which becomes available after installation through the Start Menu. From the Start Menu, the path is Start → Programs → PTP NT → AUXIO_Config.

The AUXIO_Config program defines Windows Registry settings, which define how many boards are in the system and which systems resources (I/O addresses) are required by the AUXIO boards.

Once chosen from the start menu, the AUXIO_Config main window shows as illustrated in **Figure 2-7**. The parameters in the AUXIO_Config main window are outlined in **Table 2-2**.

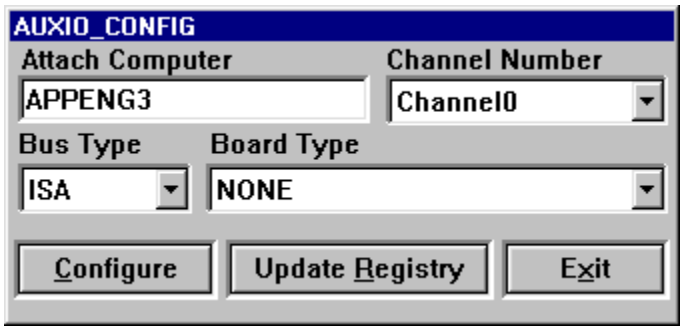


Figure 2-7: AUXIO_Config Main Window.

Table 2-2: AUXIO_Config Main Window's Parameters

Parameter	Description
Attach Computer	Displays the Computer Name for the system to be configured.
Channel Number	The AUXIO Channel number of the board to be configured. AUXIO assigns a logical channel number to each board.
Bus Type	ISA, PCI
Board Type	Type of board to be configured for the selected channel.
Configure	Displays the channel configuration dialog for the selected channel.
Update Registry	Writes the current configuration information to the Windows registry.
Exit	Exit the AUXIO_Config program.

The supported AUXIO Board Types are listed in **Table 2-3** below, which also includes the I/O port requirements for each board type. This information can be used along with the Windows Administrative Tools to select the base address for each of the AUXIO boards. The AUXIO_Config utility automatically checks to be sure that a selected address does not conflict with any other boards in the system. The configuration information for a specific board is entered using the Channel Configuration dialog shown in **Figure 2-8**. The parameters in the Channel Configuration Dialog are defined in **Table 2-4**.

Table 2-3: AUXIO_Config Available Board Types

Board Type	Description	Bus Type	Num I/O Ports
Bit Sync – Veda	Veda Bit Synchronizer	ISA	9
Viterbi Dec – Veda	Veda Viterbi Decoder	ISA	4

Board Type	Description	Bus Type	Num I/O Ports
BitSync – Aydin	Aydin PC335 Bit Synchronizer	ISA	4
PSK Mod – GDP	GDP PSK004 BPSK Subcarrier Modulator	ISA	18
Demod – Aydin	Aydin PC329 BPSK Subcarrier Demodulator	ISA	4
PSK Mod – Apogee	Apogee ISA-PSK BPSK Command Modulator (16 KHz subcarrier)	ISA	2
AT-BERT – Avtec	Bit Error Rate Tester	ISA	31
PCR2000 – Microdyne	Microdyne PCR2000 FM Receiver	ISA	4
Demod/BitSync – Avtec	Demodulator/Bit Synchronizer	PCI	N/A

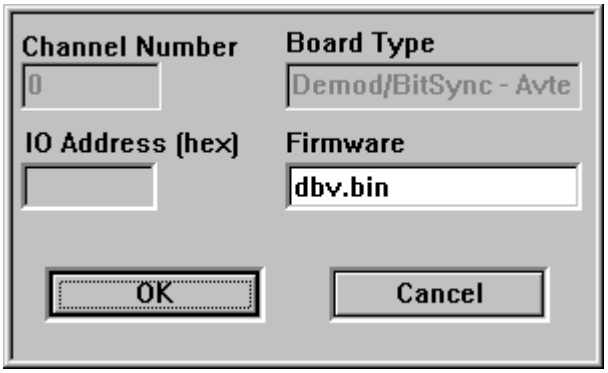


Figure 2-8: AUXIO_Config Configuration Dialog

Table 2-4: AUXIO_Config Configuration Dialog

Parameter	Description
Channel Number	Displays the channel number to be configured.
Board Type	Type of board to be configured for the selected channel.
I/O Address	Used to enter the I/O Base Address for the selected channel. This should correspond to the I/O address switch settings for the selected board.
Firmware	Displays the firmware used for that board.
OK	Updates the local copy of the configuration information and returns to the AUXIO_Config main window if no I/O address conflict is detected. An error dialog will be displayed if AUXIO_Config detects an I/O address conflict with another device in the system. Note that the new configuration information is not actually written to the Windows Registry until "Update Registry" is selected.
Cancel	Rejects any configuration changes and returns to the AUXIO_Config main window.

To configure the AUXIO device driver for a particular device (for example the Avtec AT-BERT), perform the following steps. The example below is shown for illustration purpose only, e.g. the Channel Number and IO Port Address may not be the actual ones for the board.

1) Run the AUXIO_Config program (Start → Programs → PTP NT → AUXIO_Config)

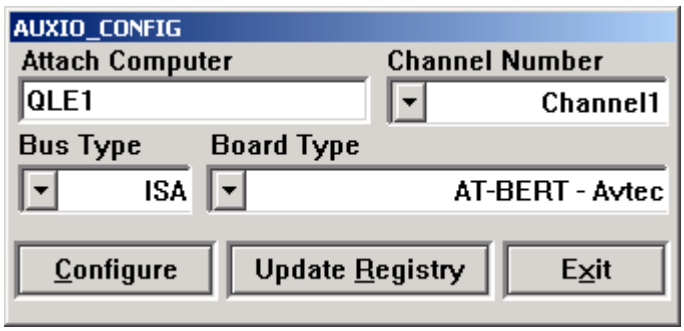


Figure 2-9: AUXIO_Config Main Window for Avtec AT-BERT

2) Refer to **Figure 2-9**. Set *Channel Number* to Channel 1.
3) Set *Bus Type* to ISA.
4) Set *Board Type* to AT-BERT - Avtec.
5) Click Configure. This will bring up the Channel Configuration Dialog window in **Figure 2-10**.

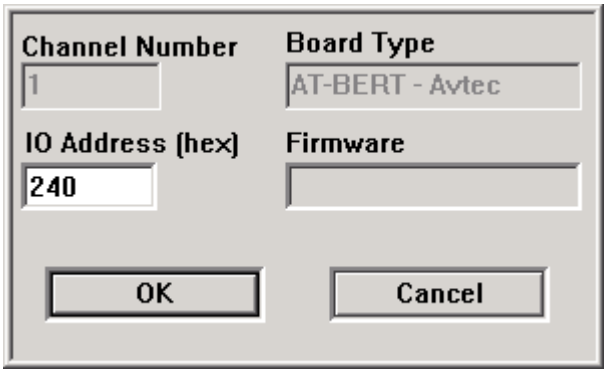


Figure 2-10: AUXIO_Config Configuration Window for Avtec AT-BERT

6) Set *IO Address* to 240**.**
7) Click OK**.**
8) Click Update Registry in **Figure 2-9**.
9) Answer *"Yes"* when asked for a confirmation (see **Figure 2-11**).

Figure 2-11: AUXIO_Config Update Registry Confirmation

Steps 2 through 9 can be repeated with a different Channel Number (Board Type and IO Address) to configure additional AUXIO boards that are in the system.

Removing AUXIO Channel Configuration

To remove an AUXIO channel from the Windows Registry perform the following steps:

1) Run the AUXIO_Config program (Start → Programs → PTP NT → AUXIO_Config)
2) Refer to **Figure 2-7**. Set *Channel Number* to the desired value.
3) Set *Bus Type* to the desired value.
4) Set *Board Type* to NONE**.**
5) Click Configure.
6) Answer *"Yes"* when asked for a confirmation (see **Figure 2-12**).

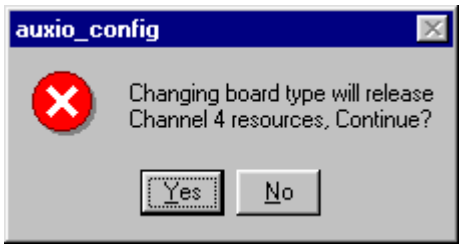


Figure 2-12: AUXIO_Config Release Channel Confirmation

7) Click Update Registry in **Figure 2-9**.
8) Answer *"Yes"* when asked for a confirmation (see **Figure 2-11**).

The Windows Registry changes will take effect the next time the AUXIO driver is loaded at boot time.

STEP 3) FRAME SYNCHRONIZER/TELEMETRY SIMULATOR (FSTS) DEVICE DRIVER CONFIGURATION

The AV_FSTS Frame Synchronizer/Telemetry Simulator driver is used by Avtec PTP for Windows to control the AT-HSIO2 and MONARCH-E high speed serial I/O boards. The Windows Registry settings for FSTS are defined using the FSTS_Config utility. The FSTS driver reads the Windows Registry settings to determine how many boards are in the system. The FSTS Windows Registry settings also define which systems resources (I/O address, IRQ line, and DMA channel) are required by the AT-HSIO2 boards.

- ✓ AT-HSIO2
- ✓ MONARCH, MONARCH E
- ✓ Apogee ISA-STG2

To run the FSTS_Config utility, select Start → Programs → PTP NT → FSTS_Config. The FSTS_Config main window will be displayed as shown in **Figure 2-13**. The parameters in the FSTS_Config main window are outlined in **Table 2-5**.

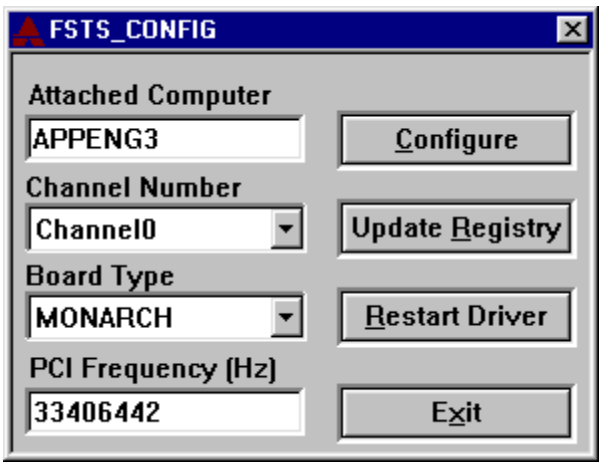


Figure 2-13: FSTS_Config Main Window

Table 2-5: FSTS_Config Main Window's Parameters

Parameter	Description
Attach Computer	Displays the Computer Name for the system to be configured.
Channel Number	The FSTS Channel number of the board to be configured. FSTS assigns a logical channel number to each board.
Board Type	Type of board to be configured for the selected channel. Valid options are AT-HSIO2, Timecode (Apogee ISA-STG2), and MONARCH. AT-HSIO2-IQ is displayed in the list, but is not a valid option for Avtec PTP for Windows.
PCI Frequency (Hz)	Frequency of the PCI bus.
Configure	1. If the selected Board Type is AT-HSIO2, then the Channel Configuration dialog will be displayed. The Channel Configuration Dialog is used to define the I/O address, IRQ line, Rx DMA channel, and Tx DMA channel that are required for the selected board. The I/O address defines the mapping between the FSTS logical channel number and the physical board. 2. If the selected Board Type is MONARCH, then a "Channel Configured" message will be displayed. There is no Channel Configuration Dialog for the MONARCH because its system resource requirements are determined at boot time using Plug-n-Play auto configuration. For systems with multiple MONARCH boards, the FSTS logical channel number to physical board location depends on the order that the PCI bus is scanned at boot. The MONARCH boards are assigned FSTS logical channel numbers in the order they are detected on the PCI bus. 3. If the selected Board Type is Timecode, then the Channel Configuration dialog will be displayed. The Channel Configuration Dialog is used to define the I/O address and mailbox channels for the selected board.
Update Registry	Writes the current configuration information to the Windows registry.
Restart Driver	Stops (sends the net stop fsts command) and starts (sends the net start fsts command) the FSTS driver.
Exit	Exit the FSTS_Config program.

AT-HSIO2 Configuration

The configuration information for a specific AT-HSIO2 board is entered using the Channel Configuration dialog shown in **Figure 2-14** below. The parameters in the Channel Configuration Dialog are defined in **Table 2-6** below.

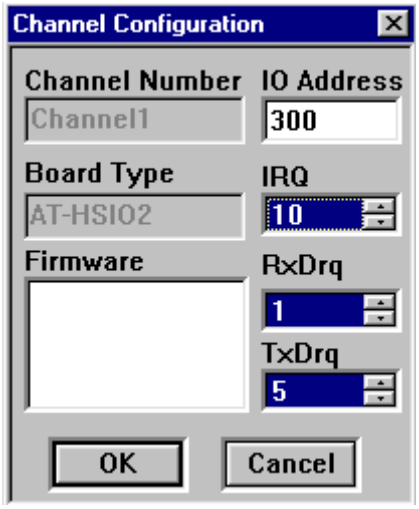


Figure 2-14: FSTS_Config Channel Configuration Dialog for AT-HSIO2

Table 2-6: FSTS_Config Channel Configuration Dialog Parameters for AT-HSIO2

Parameter	Description
Channel Number	Displays the channel number to be configured.
Board Type	Displays the board type for the board to be configured.
Firmware	Not applicable for this board.
IO Address	Used to enter the I/O Base Address for the selected channel. This should correspond to the I/O address switch settings for the selected board. This field defines the FSTS logical channel number to physical board mapping.
IRQ	Used to assign the IRQ line for the selected channel. The AT-HSIO2 board supports IRQ 9, 10, 11, 12, and 15. Only the available IRQ lines are shown. The ISA interrupt lines are not shared so each AT-HSIO2 card in the system requires its own IRQ line. The IRQ lines for ISA cards must typically be reserved in the BIOS PCI/Plug-n-Play configuration.
RxDrq	Used to assign the DMA channel to the receiver (serial input) for the selected board. Only the available DMA channels are shown. A 16-bit DMA channel (5,6, or 7) will support higher data transfer rates than an 8-bit DMA channel (0,1, or 3).
TxDrq	Used to assign the DMA channel to the transmitter (serial output) for the selected board. Only the available DMA channels are shown. If the transmitter will be used for commanding, an 8-bit DMA channel (0,1, or 3) should be used.
OK	Updates the local copy of the configuration information and returns to the FSTS_Config main window if no I/O address conflict is detected. An error dialog will be displayed if FSTS_Config detects an I/O address conflict with another device in the system. Note that the new configuration information is not actually written to the Windows Registry until "Update Registry" is selected.
Cancel	Rejects any configuration changes and returns to the FSTS_Config main window.

Important AT-HSIO2 Configuration Notes:

1) The AT-HSIO2 Base Address should be set to an address on a 32-byte boundary within the range 0x200 to 0x3FF.
2) IRQ9 should be reserved in the BIOS PCI/Plug-n-Play configuration when using the AT-HSIO2 board to prevent IRQ9 from being assigned to a PCI device.
3) DMA Channels 0 or 3 may not work for Avtec PTP for Windows systems with Tyan TITAN-III S1468 motherboards with AMIBIOS. Switching to Award BIOS corrects this problem.

The Windows Administrative Tools provide useful information about the allocation of system resources (I/O address, IRQ lines, and DMA channels). This information can be used to select the base address for each of the AT-HSIO2 boards. The FSTS_Config utility will automatically check to be sure that a selected address does not conflict with any other boards in the system.

To configure the FSTS device driver for an AT-HSIO2 channel, perform the following steps. The example below is shown for illustration purpose only, e.g. the settings shown may not be the actual ones for the board.

1) Run the FSTS_Config program (Start → Programs → PTP NT → FSTS_Config)
2) Refer to **Figure 2-13**. Set *Channel Number* to Channel 1.
3) Set *Board Type* to AT-HSIO2.
4) Click Configure. Refer to **Figure 2-14.**
5) Set *IO Address* to 300.
6) Set *IRQ* to 10.
7) Set *RxDrq* to 1.
8) Set *TxDrq* to 5.
9) Click OK.
10) Click Update Registry in **Figure 2-13.**
11) Answer *"Yes"* when asked for a confirmation (see **Figure 2-15**).

Figure 2-15: FSTS_Config Update Registry Confirmation

Steps 2 through 11 can be repeated with a different Channel Number (I/O Address, IRQ, RxDrq, and TxDrq) to configure additional AT-HSIO2 boards that are in the system.

MONARCH and MONARCH E Configuration

To configure the FSTS device driver for a MONARCH channel, perform the following steps. The example below is shown for illustration purpose only, e.g. the settings shown may not be the actual ones for the board.

1) Run the FSTS_Config program (Start → Programs → PTP NT → FSTS_Config)
2) Refer to **Figure 2-13**. Set *Channel Number* to Channel 0.
3) Set *Board Type* to MONARCH.
4) Click Configure. The channel for the Monarch board is automatically configured.
5) Click OK.
6) Click Update Registry. Answer *"Yes"* when asked for a confirmation (see **Figure 2-15**).

Steps 2 through 6 can be repeated with a different Channel Number to configure additional MONARCH boards that are in the system.

TIMECODE Configuration

The configuration information for a specific Timecode board is entered using the Channel Configuration dialog shown in **Figure 2-16** below. The parameters in the Channel Configuration Dialog are defined in **Table 2-7** below.

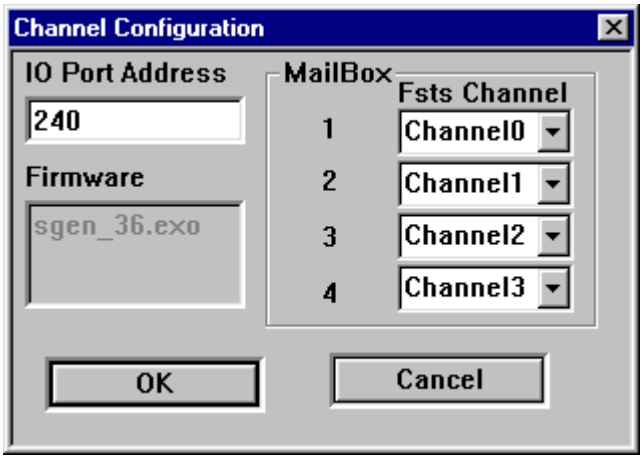


Figure 2-16: FSTS_Config Channel Configuration Dialog for Timecode

Table 2-7: FSTS_Config Channel Configuration Dialog Parameters for Timecode

Parameter	Description
IO Port Address	Used to enter the I/O Base Address for the selected channel. This should correspond to the I/O address switch settings for the selected board. This field defines the FSTS logical channel number to physical board mapping.
Firmware	Displays the firmware for the board.
Mailbox 1: Fsts Channel	Assigns each frame sync board to a physical mailbox register.

Parameter	Description
Mailbox 2: Fsts Channel	Assigns each frame sync board to a physical mailbox register.
Mailbox 3: Fsts Channel	Assigns each frame sync board to a physical mailbox register.
Mailbox 4: Fsts Channel	Assigns each frame sync board to a physical mailbox register.
OK	Updates the local copy of the configuration information and returns to the FSTS_Config main window if no I/O address conflict is detected. An error dialog will be displayed if FSTS_Config detects an I/O address conflict with another device in the system. Note that the new configuration information is not actually written to the Windows Registry until "Update Registry" is selected.
Cancel	Rejects any configuration changes and returns to the FSTS_Config main window.

To configure the FSTS device driver for a TIMECODE channel, perform the following steps. The example below is shown for illustration purpose only, e.g. the settings shown may not be the actual ones for the board.

1) Remove the Timecode board from AUXIO_Config.

   a) Start the AUXIO_Config program (Start → Programs → PTP NT → AUXIO_Config). The window in **Figure 2-17** will appear. AUXIO_Config will look for the channel that the Timecode board is configured for.

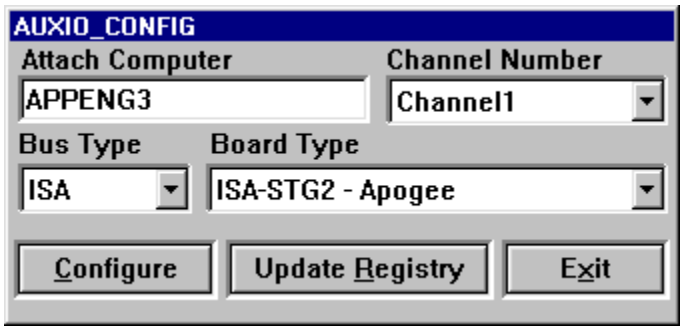


Figure 2-17: AUXIO_Config for Timecode

   b) Scan the ISA channel numbers until the Timecode board is found.

   c) Click on the Configure button and make a note of all the settings, especially the *IO Address*. Click Cancel to exit the window in **Figure 2-18**.

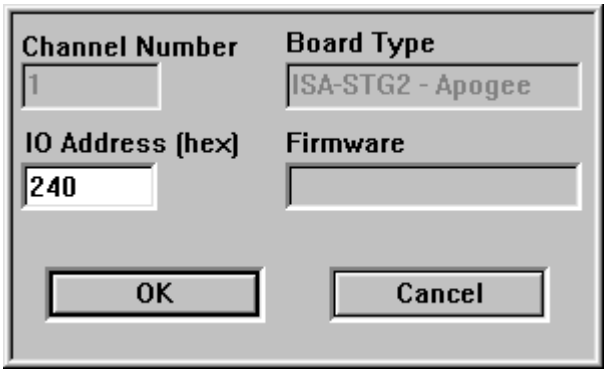


Figure 2-18: AUXIO_Config Configuration for Timecode

d) Select NONE for *Board Type* (see **Figure 2-19**) to remove the Timecode board from AUXIO_Config.

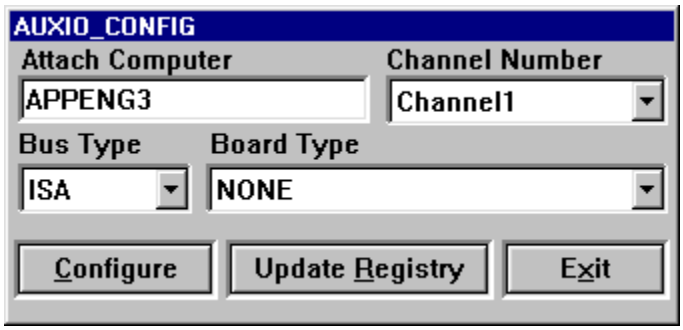


Figure 2-19: AUXIO_Config for No Board

e) Click on Update Registry and answer "*Yes*" when asked for a confirmation (see **Figure 2-11**).

f) Exit this window. This will free the resources in the Windows Registry.

2) Update/Create script files. The script file needs to be updated if you are using one to load the FPGA for the Timecode board.

a) Open the script file (or create a new one) with Notepad. The script file (filename.PTP) is located in \ptp_user\scripts directory.

b) Add or modify the line to read as follows:

```
LOADAPOGEEFPGA c:\ptp_nt\sgen_b.exo     or
LOADAPOGEEFPGA c:\ptp_nt\sgen_36.exo
```

c) Save the script file and exit Notepad.

3) Calibrate the Monarch Tic Counter Reference. The Monarch board uses the PCI bus frequency as reference for its internal hardware tic counter. The tic counter, in conjunction with the Timecode board, is used to timestamp frames.

a) Start Server and Console. Connect Console to Server.

b) Load the Monarch Tic Reference Module as shown in **Figure 2-20**.

Figure 2-20: Monarch Tic Reference Module

c) Click on the Configure button. This will bring up the dialog window shown in **Figure 2-21**. Set the *Clock Source* to Internal. When set to Internal, the Reference frequency will automatically be derived from the Monarch board's digital synthesizer. This frequency will be referenced from the external source if the clock source is selected for TTL or 422.

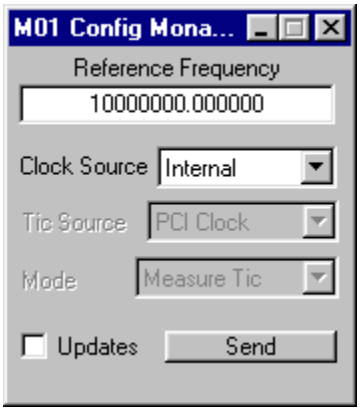


Figure 2-21: Monarch Tic Reference Configuration

d) Enable the module by clicking on the Send button. In **Figure 2-20**, click on the Status button to bring up the window in **Figure 2-22**. Let the module run for a minute until the *Average Frequency* stabilizes. Make note of the *Average Frequency* when it stabilizes. This whole number (in Hz) will be used as a reference for the PCI bus frequency in FSTS_Config.

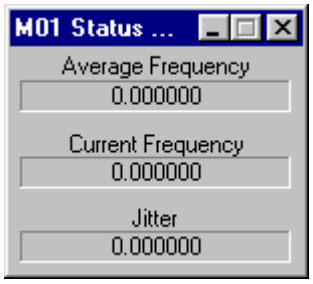


Figure 2-22: Monarch Tic Reference Status

e) Close Server and Console.

4) Run FSTS_Config program (from the Start → Programs → PTP NT → FSTS_Config)

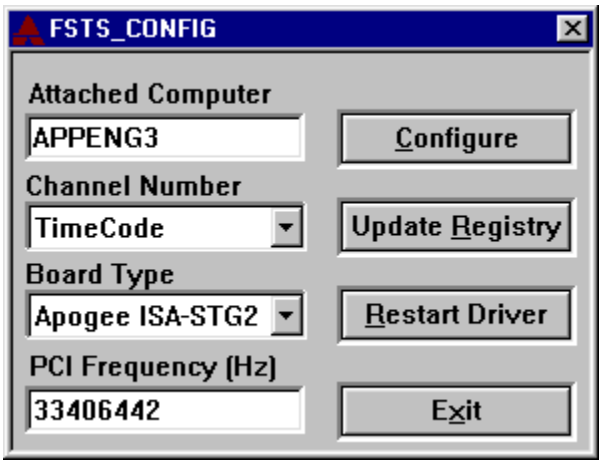


Figure 2-23: FSTS_Config for Timecode

a) Refer to **Figure 2-23**. Set *Channel Number* to Time Code.

b) Set *Board Type* to Apogee ISA-STG2.

c) Enter the PCI Frequency in Hz obtained in step 3.

d) Click Configure. The Channel Configuration Dialog will appear as in **Figure 2-16.**

e) Set *IO Port Address*. Use the same value obtained from AUXIO_Config in step 1.

f) Set the mailbox settings. The mailbox settings are used to assign each frame sync board (AT-HSIO2, MONARCH, or MONARCH-E) to a physical mailbox register on the Timecode board (up to four per time board).

g) Click OK.

h) Click Update Registry. Answer *"Yes"* when asked for a confirmation (see **Figure 2-15**).

i) Exit FSTS_Config

5) Close all programs and restart the machine.

6) Use the updated Avtec Serial Input Module. Two new options have been added to support the changes in the Timecode board. Refer to **Figure 2-24**.

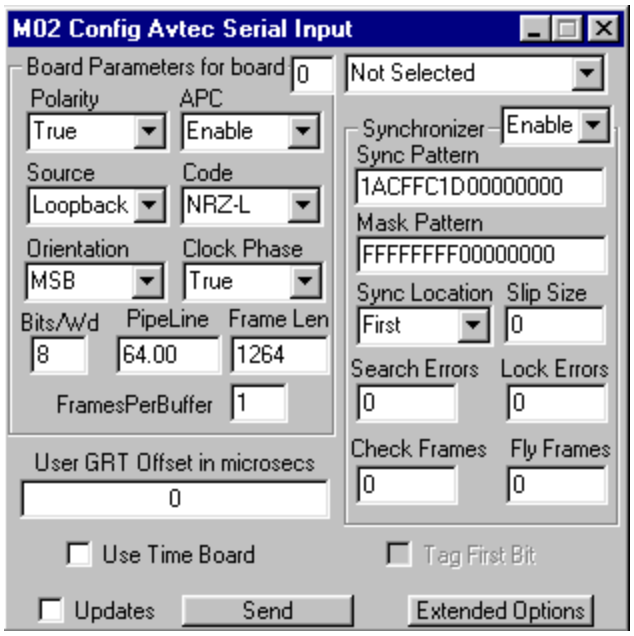


Figure 2-24: Avtec Serial Input Configuration

a) *Use Time Board:* when selected, it tells the Serial Input module to use the Timecode board and timestamp the last bit of every frame.

b) *Tag First Bit:* is only available when *Use Time Board* is selected. When enabled, time tag is recorded at the first bit of every frame.

Removing FSTS Channel Configuration

To remove an FSTS channel from the Windows Registry perform the following steps:

1) Run FSTS_Config program (from the Start → Programs → PTP NT → FSTS_Config)
2) Refer to **Figure 2-13**. Set *Channel Number* to the desired value.
3) Set *Board Type* to NONE.
4) Click Configure.
5) Answer *"Yes"* when asked for a confirmation (see **Figure 2-25).**

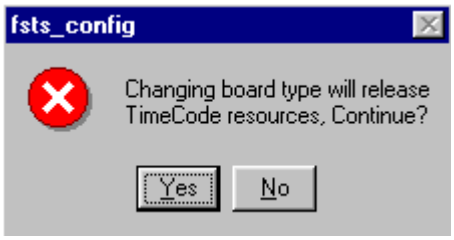


Figure 2-25: FSTS_Config Release Channel Confirmation

6) Click Update Registry. Answer *"Yes"* when asked for a confirmation (see **Figure 2-15**).

The Windows Registry changes will take effect the next time the FSTS driver is loaded at boot time. Once the FSTS device driver configuration is complete, shutdown the system and reboot Windows so that the FSTS driver will be reloaded with the new Registry settings. Check the Event Viewer (Administrative Tools) to make sure that the FSTS driver loaded properly (there should be no FSTS event messages in the Event Log). If there are any FSTS event messages, check to be sure that the I/O base address in the FSTS Config settings agrees with the switch settings on the board. If the switch settings are correct, the FSTS event message may indicate that a board in the system is not working properly and needs to be returned to Avtec for repair.

STEP 4) EEPROM UPDATE UTILITY (REV. B MONARCH E BOARDS ONLY)

Eeprom_Update is a command line program that is used to update the MONARCH (or MONARCH-E) onboard EEPROM from version 1.0 to 1.1. This update applies only to Rev. B MONARCH boards.

Since Eeprom_Update is a Console based application, each of the steps below should be executed within a command window.

1) Start the FSTS driver if it is not already started
   " net start FSTS "
2) Run the utility
   " eeprom_update "

3) The utility will scan each installed MONARCH board for type, PCB mask and eeprom revision. If an out of date combination is detected it will ask the user if the board should be upgraded. Answering "y" will proceed with the upgrade.
4) Reboot the computer to allow the changes to take effect.
5) Note that if the utility is rerun before rebooting it will incorrectly identify recently upgraded boards which will cause the utility to re-upgrade those boards. This is unnecessary, but will not cause a problem.

STEP 5) AVTEC PTP FOR WINDOWS SOFTWARE LICENSE REGISTRATION

The Avtec PTP for Windows software is licensed for installation on a single machine so the license is node locked to a particular machine. The first time that Avtec PTP for Windows (Start → Programs → PTP NT → PTP_NT) is executed after installation the display will appear as shown in **Figure 2-26**.

Figure 2-26: Avtec PTP for Windows Display before License Registration

Contact Avtec Systems with the License Key that is displayed on the second line, 408C02E6-EF3 in this example. You will be provided with a new license key that will unlock the software for the machine on which it is installed. In this example, the new license key is A069E901-B843A5A5. To register the new license type ADD LICENSE <enter> in the window. Enter the new license key at the prompt as shown in **Figure 2-27**. After the Avtec PTP for Windows registers the new license, the display will appear as shown in **Figure 2-28**.

Figure 2-27: Avtec PTP for Windows Display during License Registration

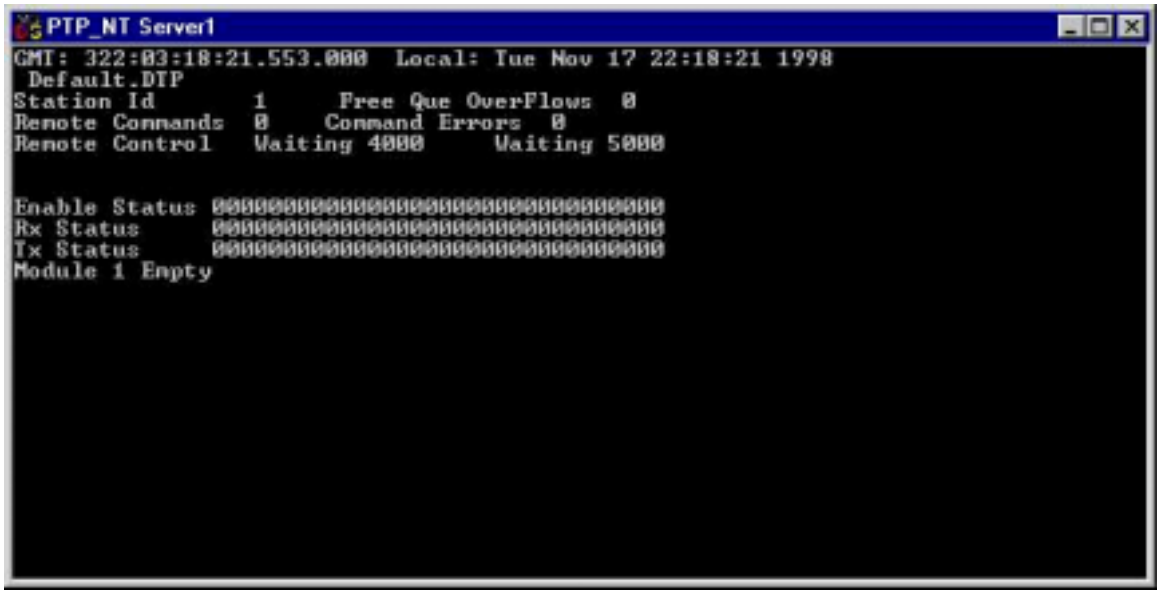


Figure 2-28: Avtec PTP for Windows Display after License Registration

The Avtec PTP for Windows stores the new license key in the USER_KEY file in the \ptp_user directory. The license must be re-registered if the Network Interface Controller (NIC) card's MAC address is changed, or the USER_KEY file is deleted, renamed, modified, or moved from the \ptp_user directory. A new registration key will also be required to install the Avtec PTP for Windows software on another machine. Changing the Computer Name or IP Address does not require re-registering the license key.

SECTION 5
UPGRADING SOFTWARE TO THE LATEST VERSION

To upgrade the Avtec PTP for Windows software to the latest version, there are some minor requirements to consider. Other than that, the basic interface and operation remain pretty much the same.

- It is recommended that the latest Service Packs are installed. For Windows NT 4.0, use Service Pack 6 (with the hot fix).
- The board that was configured under AUXIO_Config is now configured under FSTS_Config. In this case, you must remove the AUXIO_Config for that board.
- All your Desktop configurations will not be altered so you do not need to recreate them.

Chapter 3
Operation and Control

SECTION 1
INTRODUCTION

In this Chapter, operation of the Avtec PTP for Windows is explained using the Console GUI that is part of the software. The Avtec PTP for Windows can also be controlled and monitored via a custom application using RIL calls. Although top-level RIL overview information is provided in this document, refer to the AV_PTP RIL Programmer's Guide for more details of the RIL.

SECTION 2
OVERVIEW

The Console provides menu options that provide all controls that the operator needs to configure and control the Avtec PTP for Windows. From a high-level perspective, when an operator is using the Console, control and configuration revolves around "Desktops" and "Modules." The information that defines system configuration and data processing functions is saved as a Desktop (configuration file), and it is saved with an operator-defined name. The operator can reconfigure the system to the same state at any time by loading the Desktop, which is identified and loaded by name.

AVTEC PTP FOR WINDOWS CONFIGURATION

All aspects of Avtec PTP for Windows data processing are completely software configurable. Software defines which hardware is included in a data processing solution, how the hardware is set up, which software modules are included in the data processing, and how data flows from one component (hardware and software) to the next. The Desktop is the collection of information that defines a data processing solution.

The basic component of a Desktop is the Module. Each Module either configures a piece of hardware and/or provides a specific software function. A Module is a separately maintained run-time link program that performs user-specific processing in real time.

Each Module also has a unique name by which it is identified. All Modules, their functions and their names, are the same for every Avtec PTP for Windows. The Console allows the user to select a Module by name, add it to the Desktop, and then configure it to fit the requirements of that specific data processing solution. The collection of Modules required to perform a specific task, the connections between each Module (data and event exchange), and the configuration of each module are saved via the Console as a Desktop.

When a Module is added to a Desktop, a window for that module appears in the Console. Each module window provides access to all configurable information associated with that module.

AVTEC PTP FOR WINDOWS CONTROL

Once a Desktop is created, the system is configured (assuming that all supporting cables and hardware are in place). Next, the system is controlled using commands that are available on the Console Main Menu pull-down menus. An "Enable" command resets the system and makes it ready for data input. The system can be turned off (it will not recognize data input to a hardware port) if the operator issues a "Disable" command. Other controls provided by the Console main menu include zeroing of all status information and real-time control of module throughput.

The details of the Console Main and pull-down sub menus are provided in succeeding subsections of this chapter.

AVTEC PTP FOR WINDOWS STATUS

System Health

System health information is available on the Server Window, the Console title bar, and via different pull-down menus on the Console.

Data Processing Statistics

Some data processing statistics can be interpreted from the Server window. On the Server, statistics are identified by module number (each module added to a desktop is labeled with a sequential number that becomes part of the Module's window title).

Each Module that is added to a Desktop has an associated status window that reports real-time data processing statistics for that module and any hardware that it controls/monitors.

WHERE TO START

The first step of Avtec PTP for Windows operation is to load and connect the Server and Console. A Server must be running on the PC that hosts the application-specific hardware and Avtec PTP for Windows software (the Server must be running on the actual front-end system). A

GUI, or Console, must be connected to the Server via TCP/IP. The Console can run locally on the Avtec PTP for Windows or from a remote PC. To run the Console remotely, the Avtec PTP for Windows software must be loaded on the remote PC also.

Since multiple Servers can be running on the same network, the Console must be connected to the correct Server. The connection is made from the Console and is defined by Server IP address (IP of the front-end system) and Server port number. Once the connection is made, the operator can control the Server using the GUI options that are available on the Console.

BASIC STEPS TO SETUP AND CONTROL

The following steps highlight the steps to Avtec PTP for Windows setup and control:

1) Verify that external data I/O cables are correctly connected (if applicable).
2) Start Avtec PTP for Windows software: load Server and Console, and then connect them to each other.
3) Load a previously saved desktop (a desktop is the Avtec PTP for Windows name for a configuration file) or create a new desktop.
4) Enable the desktop. At enable, the system is ready for data input, or it begins the data flow for data simulation or data file read back configurations.
5) Monitor processing.
6) Disable the desktop at data flow end.
7) Before enabling the same configuration, verify that multiple enables/disables will not overwrite any data logging. Change logged data file names if necessary.

SECTION 3
GETTING STARTED

AVTEC PTP FOR WINDOWS SERVER AND CONSOLE STARTUP

Avtec PTP for Windows Server and Console startup can either be manual (the user must start the software) or turnkey (software starts automatically at PC boot). Manual mode is the default configuration. Software must be added to the Startup folder for turnkey mode. In either configuration, the Server must be started before the Console.

Multiple Servers can be loaded/executed by executing *ptp_nt.exe* multiple times. Each Server is numbered incrementally from 1 to n (where n is the number of Servers loaded). The Server name appears in the Server window title bar. For example, the first Server loaded is titled "PTP_NT Server 1." The first Server is assigned control port 4000, the second Server is assigned control port 4001, and port assignment continues to port 400n.

Manual Mode

The default system configuration requires Manual startup of the Avtec PTP for Windows software. Manual startup is a three-step process:

step 1: Start the Server

step 2: Start the Console

step 3: Connect the Console to the Server.

To connect the Console to the Server, use the *System* pull-down menu that is available from the Console Window. Go to the *Connect Control To* option.

If multicast is enabled, *Connect Control To* lists the available Servers by IP address and port number. The user selects the correct Server from the list and issues a connect command. If a Server is already connected to a Console, then the connection shows in the *Connect Control To* list and any attempt to connect to that Server will fail.

If multicast is disabled, then the *Connect Control To* dialog window does not list any Server addresses even if they are available on the network. In this case, the operator must type the Server IP address and the port number into the dialog box, and then issue *SEND*.

Once connected, the connection information appears in the Console title bar. If the connection fails for any reason, the failure is also reported in the Console title bar.

Default configuration of Avtec PTP for Windows allows the Server and Console to be started via the Start Menu. Depending on how the system is configured, the Server and Console may also be available from the PC desktop. Server and Console are always available from the command line window. **Table 3-1** details startup from the three possible paths.

Table 3-1: How to Manually Start Avtec PTP for Windows Software

	Server	Console
Desktop	Double click on PTP NT icon	Double click on Console icon
Start Menu	Follow the path to PTP_NT: Start → Programs → PTP NT → PTP_NT	Follow the path to Console: Start → Programs → PTP NT → Console
Command Prompt	Enter (where c is the root directory): c:\ptp_nt\PTP_NT	Enter (where c is the root directory): c:\ptp_nt\console

Turnkey Mode

To configure the Avtec PTP for Windows to automatically execute at system startup, copy the PTP NT and Console icons into the "Startup" folder. Each time the system is powered down and restarted, the Avtec PTP for Windows automatically opens to a user-specified desktop. To specify a particular desktop to be opened at system startup, use a script file.

The computer may be optionally configured to login automatically on boot up. To setup the computer to login automatically on boot up, perform the following steps.

1) Add a user called AVPTP_USER with password AVPTP_USER. Add AVPTP_USER to the Power Users or Administrators group.

2) Run Regedit.exe (in the \Winnt directory)
3) Go to
   hkey_local_machine\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
4) AddValue:

AutoAdminLogon	REGSZ	1
DefaultPassword	REGSZ	AVPTP_USER

5) ModifyValue:

DefaultUserName	REGSZ	AVPTP_USER

To bypass the auto logon, hold down the Shift key during shutdown and reboot. This will allow you to logon as another user.

To disable the auto logon, modify the *AutoAdminLogon* value to be 0 or leave the value blank, and change the *DefaultPassword* value to be blank.

The user can write a script file *(use file name script*.ptp, where * refers to the Server number)* and the Server will automatically execute the commands in that script file when you open a Server window of the same number. See **Section 5**, for more detail.

All Servers will look for the file "script.ptp" and run this script file. The software will then look for the file "script*.ptp" if it exists, where * refers to the Server number. Lastly, if the operator has a working directory set, the software will look for the "script.ptp" file in this folder.

LOGGING IN/DEFAULT PASSWORDS

The Avtec PTP for Windows User must belong to the Power Users or Administrators group. At a minimum, it must have the "Increase scheduling priority" right. This right is granted by default to the Administrators group, so if the Avtec PTP for Windows User belongs to this group, it will automatically inherit this right.

SERVER WINDOW DESCRIPTION

Figure 3-1 illustrates a Server Window. This is the window that appears as soon as the Server application is loaded. If at startup, the Server window reverts to Decom mode, then the software license must be registered. Refer to **Chapter 2, Section 3, Step 5** for the details of software registration.

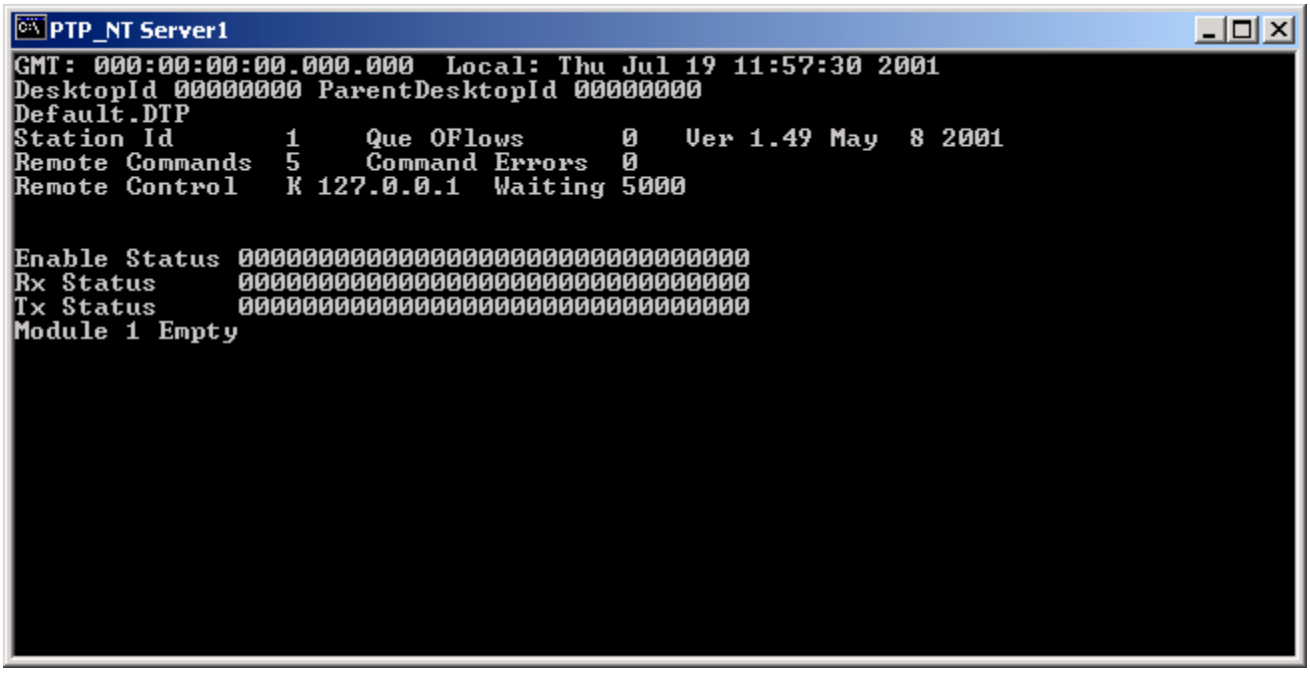


Figure 3-1: Server Window

Server Window Status

The Server window contains 13 information fields, defined in **Table 3-2.** When modules are loaded, the bottom part of the PTP Server Window displays status for a particular module. Modules are identified by module number.

Table 3-2: Server Window Status

Status	Definition
Title Bar	Identifies Server number
GMT	From time board if loaded, from system time otherwise
Local	Current date and time
/Default.DTP	Fully qualified desktop name
Station Id	User-assigned Station Id
Free Que Overflows	Number of failed buffer requests
Remote Commands	The Avtec PTP for Windows provides two socket ports for remote control (4000 and 5000). This field shows if a control client is connected. If no client is connected then *Waiting 4000* or *Waiting 5000* is displayed. If a client is connected then the client's IP address is displayed.
Command Errors	Total number of remote control commands received
Remote Control	Either DISConnected or Connected. If connected, the client's IP address is displayed.
Enable Status	Bit-mapped module enable flags: 1 = enable, 0 = disable; Module 1 is left-most bit, module 32 is right-most bit
Rx Status	Bit-mapped activity flag based on programmable activity monitor time; 1= module received a buffer within the activity monitor time out

Status	Definition
Tx Status	Bit-mapped activity flag based on programmable activity monitor time; 1= module transmitted a buffer within the activity monitor time out
Module 1 Empty	If a module is loaded, this field name is replaced with module-specific data for the remainder of the display

Changing Status Display

You can switch the display among loaded modules by typing the command:
view module_number <CR>

where module_number is the number (1,2,3, ...) of the module to display.

You can quit the Avtec PTP for Windows application by typing command:
quit <CR>

Appendix A lists all commands that can be executed from the Server application.

CONSOLE WINDOW DESCRIPTION AND MENU OPTIONS

The Console Window is your desktop for creating data processing solutions with the available modules.

Console Main Menu

The Console window appears as soon as the Console application is loaded. **Figure 3-2** illustrates a Console window. The Console window Title Bar displays the status of the Console application. The Console Main Menu is located directly below the title bar and consists of the following options: *System, Remote Desktop*, *Remote Module,* and *Local Console*. The Main Menu is the Avtec PTP for Windows command center. Each option on the main menu provides a pull down menu.

The Console shown in **Figure 3-2** is not connected to a Server. When not connected, if any of the Main Menu pull-down menus are accessed, most of the options are grayed out and unavailable for use. The options become available as soon as the Console is connected to a Server. To connect, use the *System Menu/Connect Control To* option.

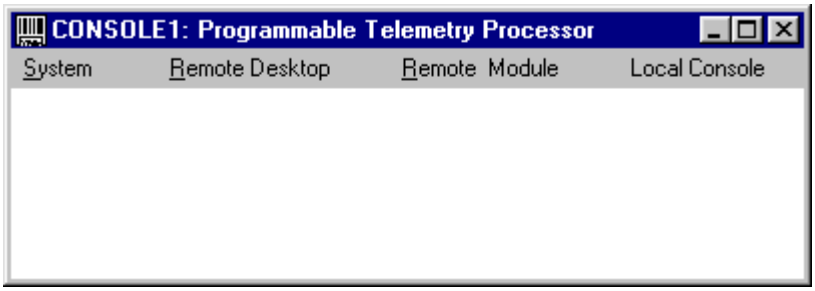


Figure 3-2: Console Window at Startup

Console Status

Once the Console has connected to a Server, the Console window status changes and if any modules are already loaded in the Server (before connection is made), the Console Window displays the dialogs for those modules as soon as the connection is made. Once the Console is connected to a Server, the Avtec PTP for Windows can be controlled using the menu options. **Figure 3-3** illustrates a Console window that is connected to a Server but has no modules loaded.

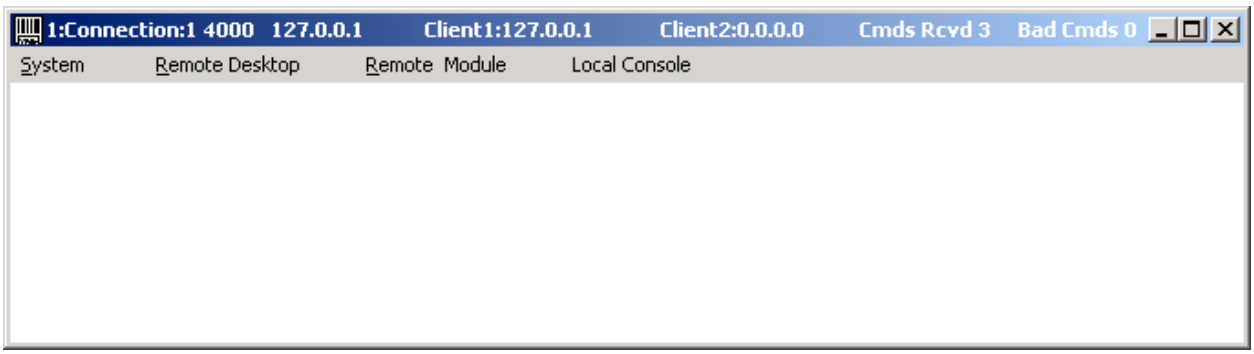


Figure 3-3: Console Window after Connecting to a Server

The Console title bar contains six information fields:

- *Not Connected* or *1:Waiting:1* or *1:Connection:1* or *Status* – Indicates the Console-to-Server status
- *4000* – Station ID of the Server, (the TCP control socket port number)
- *127.0.0.1* – Internet address of the Server
- *Client 1:127.0.0.1* – Internet address of the Console client
- *Client 2: 0.0.0.0* – No control client on record control port
- *Cmds Rcvd* – Counts the commands received by the Server
- *Bad Cmds* – Counts the bad commands received by the Server

"System" Pull Down Menu

The *System* option pull down menu, shown in **Figure 3-4**, provides top-level functions that include connection of that Console to a Server. The System menu options are defined in **Table 3-3**:

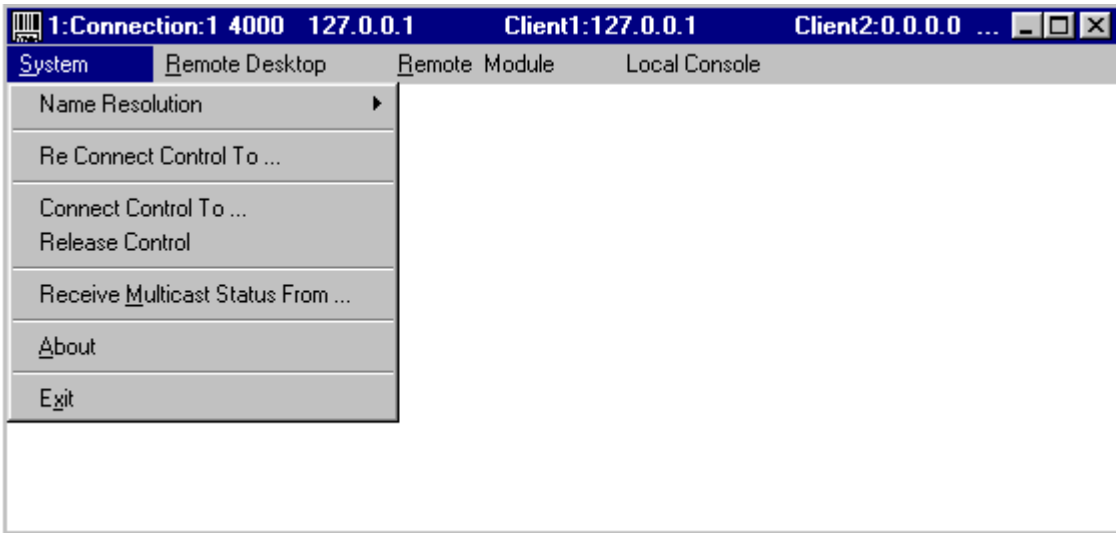


Figure 3-4: "System" Pull Down Menu

Table 3-3: "System" Pull Down Menu Options

Feature	Function Key(s)	Description
Name Resolution	→	Controls the use of name resolution within the Console application. If a domain name Server is available, Name Resolution can be enabled to identify Servers by name. Otherwise, Servers must be identified by IP address.
Re Connect Control To		Reconnects to a previous Server.
Connect Control To		Connects to a Server. Displays a dialog box with a list of Servers on the network.
Release Control		Disconnects from a Server.
Receive Multicast Status From		Receives status from a Server.
About		Provides version number for the Console and for Server.
Exit		Exits Avtec PTP for Windows

Name Resolution

Name Resolution under the *System* menu controls the use of name resolution within the Console application. As shown in **Figure 3-5**, an arrow to the right of a selection indicates that it includes a sub-menu. Moving the mouse to the highlighted item and clicking the left button activates the sub-menu, which will appear either to the right or left of the pull down menu.

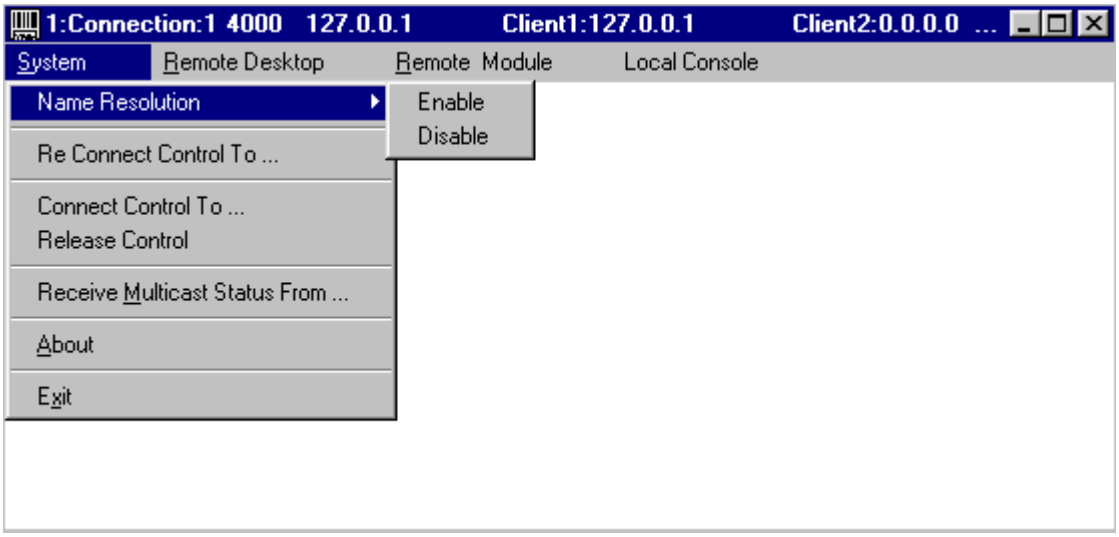


Figure 3-5: "Name Resolution" Sub-menu

Connect Control To…

Once Console is loaded, it may be connected to a Server. The *Connect Control To* function provides the connection controls. **Figure 3-6** shows the *Connect To* window. If multicast is enabled, the *Connect To* window lists the Servers that are currently available on the network. The Console identifies Servers by their IP addresses. If multicast is disabled or if the target Server is on a network that does not support IP multicast, then the operator must manually enter the correct Server IP address and connection port in the dialog window.

Note: You must have a Server running on your network in order to connect your Console to Avtec PTP for Windows. You must connect the Console to a Server even if both applications are running on the same machine.

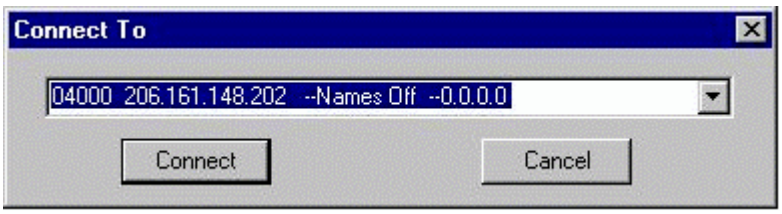


Figure 3-6: Connect To Window

Note: The Server availability "list" freezes when you select *Connect Control To*. The *Connect To* window will not notify you if other Servers come online after you select *Connect Control To*.

Each list entry in the *Connect To* dialog box contains the following fields:

- *04000* – Control socket port number for the Server (Station ID)
- *206.161.148.202* – IP address for the Server

- *Names Off* – Indicates that name resolution is disabled. This field would show the host name for the PTP Server if name resolution is enabled.
- *0.0.0.0* – IP address of the client that currently controls the Server. An address of all zeroes indicates that no client is controlling the Server.

There is a file under the \ptp_user\configs folder called "UserConnections.txt" that lists all the Servers' control socket port numbers and IP addresses. The Avtec PTP for Windows software reads this file to populate the fields in the pull down menu of **Figure 3-6.** Here is an example of what the "UserConnections.txt" file looks like. The first column lists the control socket port numbers. The second column lists their corresponding IP addresses.

```
4000     206.161.148.202
4001     198.102.100.10
4002     200.192.160.25
```

Release Control

The Console can change from one Server connection to another. To change Servers, pull down the *System* menu and select *Release Control*. Once released, a new connection can be made with *System/Connect Control To*.

Receive Multicast Status From…

To monitor the status of a Server, select *System Menu*/*Receive Multicast Status From*. This mode will display the periodic broadcast status from the Server.

"Remote Desktop" Pull Down Menu

Desktop design and control (enable, disable, etc) are done via the *Remote Desktop* pull down menu. **Figure 3-7** illustrates the *Remote Desktop* pull down menu options, which are defined in **Table 3-4**:

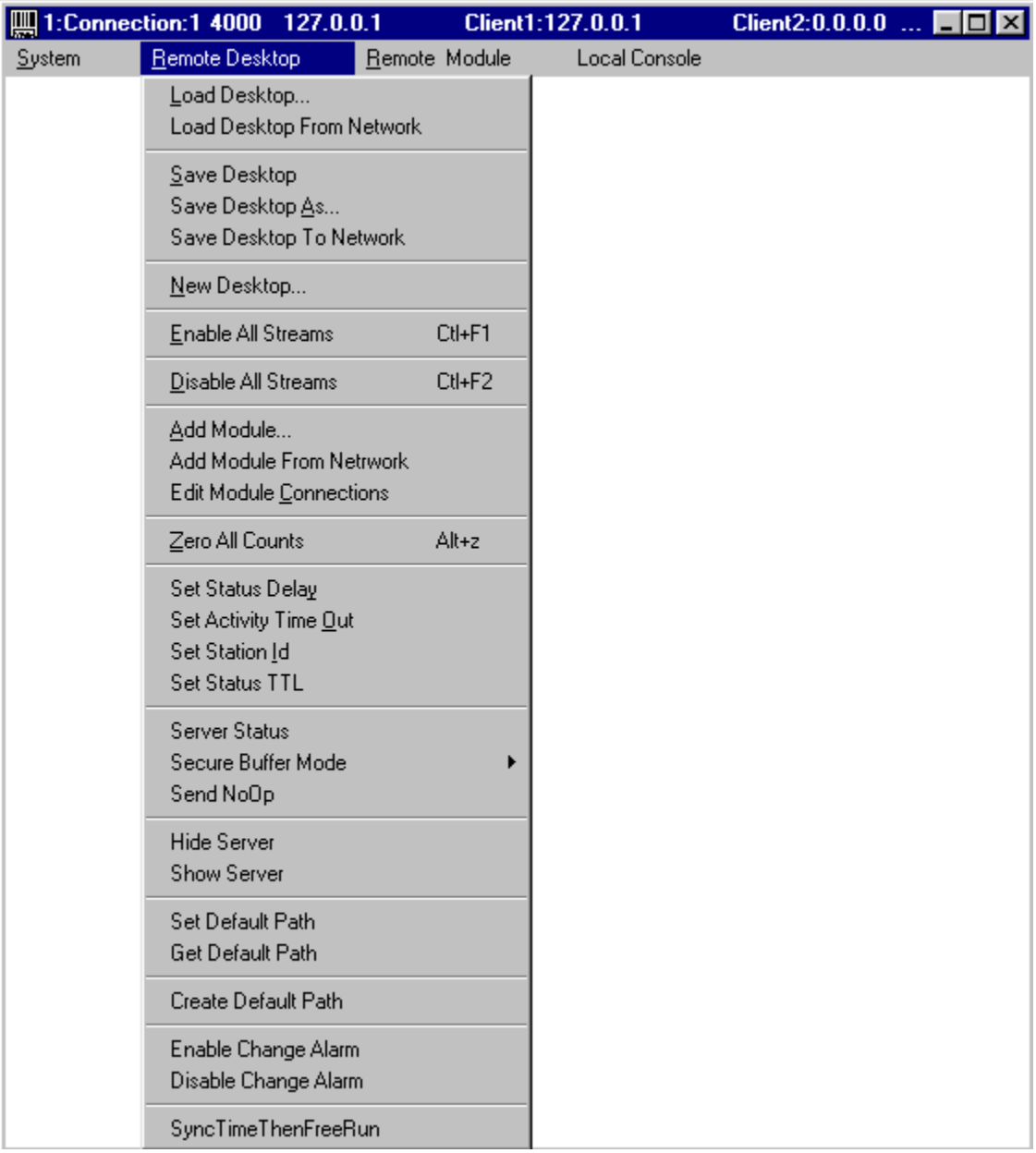


Figure 3-7: "Remote Desktop" Pull Down Menu

Table 3-4: "Remote Desktop" Pull Down Menu Options

Feature	Function Key(s)	Description
Load Desktop…		Loads an existing desktop that was saved from an earlier operation. This command searches the \ptp_user\desktops folders for available desktops. It does not provide folder exploring capabilities. If the desktop was saved with the modules enabled, it will resume updating all active data fields upon opening.
Load Desktop From Network		Same as *Load Desktop*, but allows the user to search directories on the LAN for desktop files (provides folder exploring).

Feature	Function Key(s)	Description
Save Desktop…		Saves the current desktop to a .DTP file. Uses the current desktop file name shown on line 2 of the Server display.
Save Desktop As…		Opens Save Desktop Directory/File Window to save the current desktop to a new .DTP file in the \ptp_user\desktops folder.
Save Desktop To Network		Same as *Save Desktop As*, but allows the user to save the desktop anywhere on the local computer or LAN.
New Desktop…		Generates a completely new desktop, returning the operator to the Avtec PTP for Windows Opening Window.
Enable All Streams…	Ctl+F1	Enables all of the modules on the desktop.
Disable All Streams…	Ctl+F2	Disables all of the modules on the desktop.
Add Module…		Opens Load Module Directory/File Window for user to select a module to add to the desktop from the \ptp_nt\modules folder on the local machine.
Add Module From Network		Same as *Add Module*, but allows the user to select modules from anywhere on the local computer or LAN.
Edit Module Connections		Edits the data and event connections for the modules loaded on the desktop.
Zero All Counts	Alt+z	Zeroes counters in all windows, active or not.
Set Status Delay		Sets the time delay in milliseconds (msec) between periodic status updates by the Server. The default value is 1000 msec.
Set Activity Time Out		Sets time out the interval for the Server activity monitor. The activity monitor checks for data activity in the loaded modules and displays the results in the bitmap Rx status and Tx station in the Server. The default value is 1000 msec.
Set Station Id		Sets the Station ID for the Server
Set Status TTL		Sets the Time-to-Live (TTL) (i.e. number of hops) for a status message before the message is killed. This is a global command and applies to all module status messages. The default status TTL is 1.
Server Status (see **Figure 3-8**)		Displays the Server status. Reports the current desktop name and location. Reports the current time, and reports each modules enabled and data transfer status.
Secure Buffer Mode		Can be enabled or disabled. When enabled, whenever a module sends one data stream to multiple modules, creates a copy of the data stream for each module. When disabled, whenever a module sends one data stream to multiple modules, the same file pointer location is given to all modules.
Send NoOp		Sends No Operation message to check if anyone is alive.
Hide Server		Hides the Server. It only shows up under Processes in Task Manager.
Show Server		Brings back the hidden Server. Requires a connection to the Server.
Set Default Path		Sets default path to the working directory and appends \ptp_user\… to the path. Assumes the working folder has already been created.
Get Default Path		Gets the current working directory.

Feature	Function Key(s)	Description
Create Default Path		Creates the working folder.
Enable Change Alarm		Enables the Change Alarm dialog box.
Disable Change Alarm		Disables the Change Alarm dialog box.
SyncTimeThenFreeRun		Assumes a time board is connected. Synchronizes time board's time to station time (IRIG). If fails, sets time board's time back to PC time. After the test is finished, the syncmode is put in free run mode.

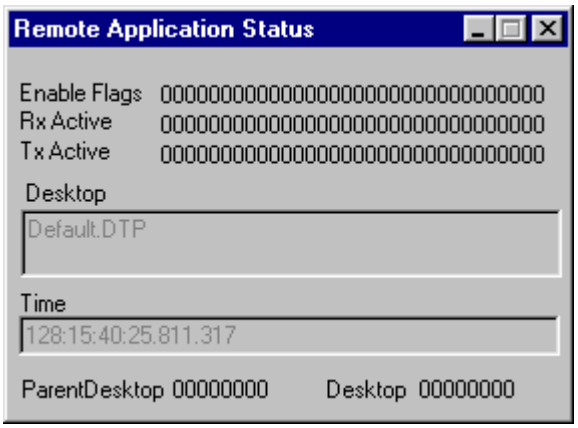


Figure 3-8: Server Status

Change Alarm

The Change Alarm is used to notify the user connected to the Avtec PTP for Windows that his local desktop has been modified remotely elsewhere. When the local Console senses that its desktop has been changed by another Console elsewhere (on a different control port, for example), the Change Alarm window will appear as shown in **Figure 3-9** notifying the user what changes have occurred.

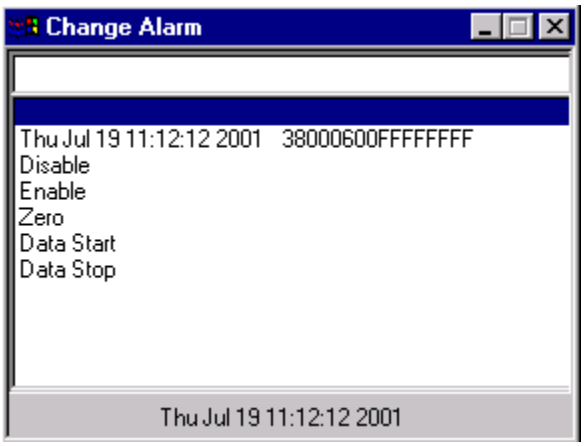


Figure 3-9: Change Alarm

"Remote Module" Pull Down Menu (Working with Highlighted Modules)

The *Remote Module* pull down menu provides control on a per module basis. When implemented, a *Remote Module* pull down menu option performs the selected operations on the window or module that is currently highlighted. **Figure 3-10** illustrates the *Remote Module* menu. **Table 3-5** defines the Options available in the *Remote Module* menu.

The term *highlighted* refers to the "active" window and is indicated by a darker title bar. A module is highlighted by clicking on the window with the left mouse button (the Title Bar turns active when the module is highlighted). Once a module has been highlighted, any of the options available from the *Remote Module* pull down menu can be implemented and it affects the highlighted module only.

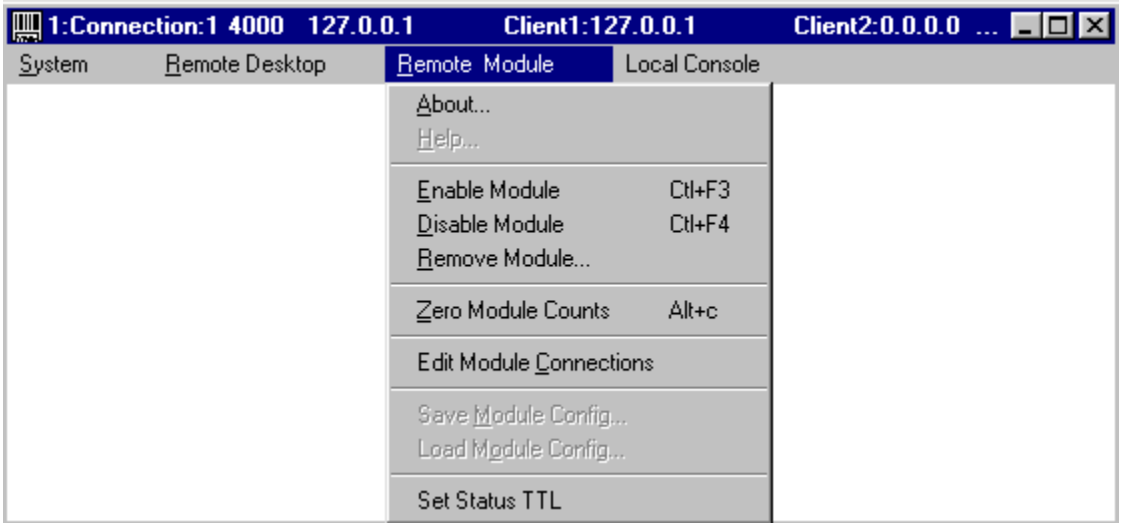


Figure 3-10: "Remote Module" Pull Down Menu

Table 3-5: "Remote Module" Pull Down Menu Options

Feature	Function Key(s)	Description
About…		Not implemented for all modules.
Help…		Not implemented for all modules.
Enable Module	Ctl+F3	Enables the selected module.
Disable Module	Ctl+F4	Disables the selected module.
Remove Module…		Removes the selected module.
Zero Module Counts	Alt+c	Zeroes the status counts for the selected module.
Edit Module Connections		Provides a window that allows the operator to edit the data and event connections for the selected module.
Save Module Config…		Currently not implemented.
Load Module Config…		Currently not implemented.
Set Status TTL		Sets the Time-to-Live (TTL) (i.e. number of hops) for a status message before the message is killed. This applies only to the selected module. The default TTL is 1.

"Local Console" Pull Down Menu

Figure 3-11 illustrates the *Local Console* menu. **Table 3-6** defines the Options available in the *Local Console* menu.

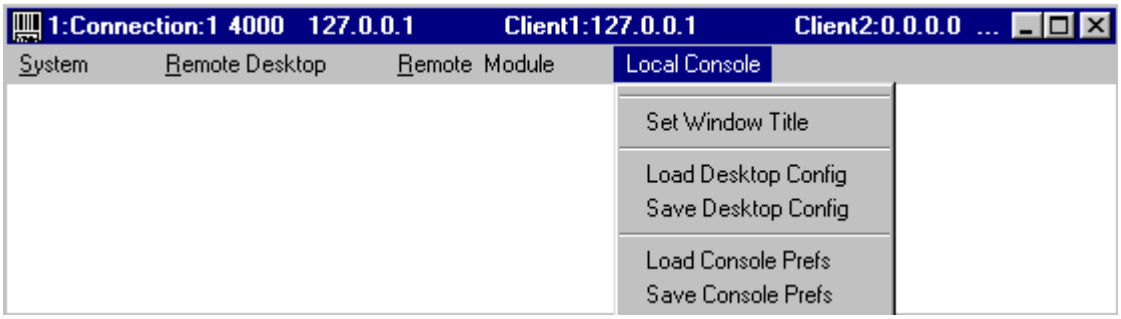


Figure 3-11: "Local Console" Pull Down Menu

Table 3-6: "Local Console" Pull Down Menu Options

Feature	Function Key(s)	Description
Set Window Title		Changes the title of the highlighted module. When the parent module's title is changed, all of its child module's titles are changed also.
Load Desktop Config		Currently not implemented.
Save Desktop Config		Saves the desktop configuration to the local path. The local console path is always \ptp_user\…
Load Console Prefs		Loads the console preferences from the local path. The local console path is always \ptp_user\…
Save Console Prefs		Saves the console preferences to the local path. The local console path is always \ptp_user\…

SECTION 4
WORKING WITH MODULES AND DESKTOPS

AVTEC PTP FOR WINDOWS DESKTOP

To create a desktop, the operator is responsible for learning the functions of the available Modules, and then selecting the correct Module combination to meet the data flow requirements. If a Module is associated with a piece of hardware, it cannot be used in the system configuration unless that piece of hardware is installed.

In order to create a desktop, it is vital to know what a Module expects for input, the processing that the Module performs, and what the Module outputs. Module I/O is not just data. Some Modules generate status information, generate annotation, and some also generate an event output or require and event input. Usually, a Module generates an event to mark the end of a data

unit. A Module can require an event input to drive data flow through the Module or trigger data output. Refer to the module descriptions provided in **Chapter 4** for details of each Module's processing and I/O.

Module Window

To create a Desktop, the operator selects (adds) Modules from the available list. As a Module is added, a window appears in the Console for that Module. The title bar of the window provides the module name and also shows an assigned module number.

Each Module window provides access to the control, configuration, and status that are associated with that module. Buttons on the Module window open configuration, status, and control windows.

To configure a Module, the operator clicks the "Configure" button which opens the configuration window and changes the available parameters and processing options to meet the data flow requirements. Once a Desktop is enabled and data flows through the Module, the operator can monitor real-time statistics through the status window that is accessed by clicking on the "Status" button. Several Modules include a "Command" button on the Module window. For these modules, real time commands can be issued via the command window. More details are discussed in **Chapter 4, Section 1**.

Module Number

Module numbers are key to making connections (data and events) between modules because connections are made by module number and not by module name.

A module number (showing in the module title bar once it is added to a desktop) is assigned to each Module added to a desktop. The module number is desktop dependent. Module numbers are assigned to Modules sequentially based on the order in which each is added to the desktop, and starting at Module number one.

The first Module added to a desktop appears on the Console in the top-left corner. Each additional Module added appears beneath the previous Module until the Console's vertical space is filled, at which point a second column of Modules begins. Based on how Modules appear on the Console and how module numbers are assigned, the operator would expect to see sequential numbering of modules starting with the top left and reading down. However, exceptions to sequential numbering do occur if Modules are removed from a desktop. Also, if Modules are removed and then others added, the order on the screen no longer reflects the order in which Modules were added to the desktop.

If a Module is removed from a desktop, the numbers assigned to the other Modules on the Console do not change. This means that there is a gap in the sequential numbering. If another module is added after a module has been removed, the most recently added module will assume the removed Module's number. Added modules assume the first missing module number sequence gap until all gaps are filled, and then numbering resumes after the highest module number.

Module Connections

The Desktop consists of Modules that need to be connected together to define the data flow path. Module I/O may be either data or events.

The Console provides an "*Edit Module Connections*" menu option through which the operator sets up the data and event flow through the modules.

AVAILABLE MODULES AND FUNCTIONAL OVERVIEW OF EACH

The Avtec PTP for Windows provides four types of modules:

1) Auxiliary I/O: Auxiliary I/O Modules control and monitor auxiliary I/O boards such as a Bit Synchronizer, a BPSK modulator, a BPSK demodulator, or a Time Code Processor. These modules do not transfer real-time data streams into or out of the system. Instead, the module passes control and status information between the CPU and the auxiliary I/O board.
2) Data I/O: Data I/O Modules have an associated piece of hardware. Each module controls a data I/O device in the system, such as network interfaces, file devices, and the AT-HSIO2 or MONARCH-E boards. The Data I/O Modules transfer real-time data streams into or out of the system using the application-specific interfaces and standard PC peripherals
3) Data Processing: Data Processing Modules do not have an associated piece of hardware. The module provides a software function that is performed on data as it flows through the module.
4) Encapsulation/Extraction: These are data formatting modules that either create a data format or extract data from a format. In many cases, the modules provide spacecraft specific headers that can be added to a data unit.

Table 3-7 lists the available modules and their functions.

Table 3-7: Available Modules and Their Functions

Data I/O Modules		
Name	**DLL**	**Description**
Avtec Rate Adjust	AvtecRateAdjust.dll	Serial transmit to AT-HSIO2 or MONARCH-E PCM Simulator with buffering to allow module output data rate less than module input data rate.
Avtec Serial Input	AvtecSerialInput.dll	Serial receive from AT-HSIO2 or MONARCH-E Frame Synchronizer
Avtec Serial Output	AvtecSerialOutput.dll	Serial transmit to AT-HSIO2 or MONARCH-E PCM Simulator
CPU Timer	CPUTimer.dll	Provides an event output at a user-configurable rate. Useful for simulating output end-of-frames when no hardware is available in the system.
Com Port Transceiver	Comport.dll	Allows sending and receiving of data via PC's Com Ports.

Dump to Line Printer	DumpToLP.dll	Allows formatted data output to a printer or file.
File PlayBack	FilePlayBack.dll	Replays a previously recorded data file
File PlayBack Version 2	FilePlayBackVer2.dll	Replays a previously recorded data file. Supports file sizes > 4 GB. Optimized for dual processors.
File Recorder	FileRecorder.dll	Records the output of a module to disk
File Recorder Version 2	FileRecorderVer2.dll	Records the output of a module to disk. Supports file sizes > 4 GB. Optimized for dual processors.
File Spooler (Spooler)	Spooler.dll	Allows spooling of data to a file for playback at a slower/faster rate.
Network Sockets (Sockets)	NetworkSockets.dll	Sends or receives data over a network
Network Sockets Version 2 (Sockets Ver 2)	NetworkSocketsVer2.dll	TCP Server mode only socket with support for multiple TCP Clients.
Auxiliary I/O Modules		
Name	**DLL**	**Description**
Apogee PSK Modulator	ApogeePsk.dll	Configures and controls the Apogee ISA-PSK BPSK Subcarrier Generator
Avtec Demod Bit Sync with Viterbi	Avtec DemodBitSyncViterbi.dll	Setup, Control, and Monitor Avtec's Demod/Bit Sync/Viterbi Board
Aydin Bit Sync	AydinBitSync.dll	Configures and controls the Aydin PC335 Bit Synchronizer
Aydin Bit Sync with Viterbi	AydinBitSyncViterbi.dll	Controls Aydin PC 335V Card
Aydin PSK Demodulator	PSKDemod.dll	Configures and controls the Aydin PC329 PSK Demodulator
ATM	ATM.dll	Controls Forerunner LE ATM Adapter OC-3 board
CMOD	Cmod.dll	Provides control of the ATSC built Command Modulator board using ATSC supplied software
DCS Demod	DCSDemod_BS.dll	Controls the Data Capture Systems (DCS) Demodulator, BitSync and Viterbi unit
GDP PSK Modulator	PSKMod.dll	Configures and controls the GDP PSK004 PSK Modulator
Microdyne PCR2000	PCR2000.dll	Configures and controls the Microdyne PCR2000 Receiver/Demodulator
National Instruments GPIB	GPIB.dll	Configures and controls the National Instruments GPIB Interface Board
Odetics GPSync-ISA Time Board	OdeticsTime.dll	Configures and controls the Odetics GPSync-ISA Time Board
PSK Summer	PSKSummer.dll	Controls ATSC PSK Summer Card
Veda Bit Sync	VedaBitSync.dll	Configures and controls the Veda bit synchronizer

Data Processing Modules		
Name	**DLL**	**Description**
8 Bit Sorter	Sorter8bit.dll	Sorts packets based on 8 bit value in a user-specified byte location
16 Bit Sorter	Sorter16bit.dll	Sorts packets based on 16 bit value in a user-specified byte location
BCH Encoder	BCHEncoder.dll	Implements the Coding layer defined in the CCSDS Telecommand
BCH Decoder	BCHDecoder.dll	Implements the inverse of the Coding layer defined in the CCSDS Telecommand
Best Source Select (BSS)	BestSourceSelect.dll	Used to compare identical incoming streams and output "best" stream based on frame count
Bit Density Correction	BitDensityCorrection.dll	Guarantees signal transitions in a bit stream by XORing stream with 2047 pseudo-random pattern
Bit Error Rate Tester	BitErrorRateTester.dll	Sends (and receives and verifies) a pseudorandom 2047-bit pattern to quality check a communications link
CLCW Processor	CLCWProcessor.dll	Extracts the Command Link Control Word from a user-defined offset in the telemetry stream
CCSDS Bit Stream Processor	CCSDS BitStreamProcessor.dll	Performs CCSDS AOS Bitstream Service
CCSDS Virtual Channel Processor	VirtualChannelProcessor.dll	Processes and displays statistics on CCSDS VCDUs
CCSDS Virtual Channel Processor Version 2	VirtualChannel ProcessorV2.dll	Processes and displays statistics on CCSDS VCDUs. Monitors and reports Master Channel sequence count errors and missing frame counts.
CCSDS Virtual Channel Simulator	CCSDSVirtualChannel Simulator.dll	Provides a user-configurable source of CCSDS VCDUs
Command Verify	CommandVerify.dll	Compares uplink data to radiated data to validate data radiated from the antenna feed horn
Convolutional Encoder	ConvolutionalEncoder.dll	Convolutional encodes any input stream
Decom	Decom.dll	User defined display of TDM, CCSDS parameters.
Delay	Delay.dll	Delays a data stream by a certain number of milliseconds
Error Inject	ErrorInject.dll	Induces errors in a data stream
Gather Scatter	GatherScatter.dll	Extracts user-specified bytes from a data stream
In-Line UNZip	InLineUNZip.dll	Performs data decompression
In-Line Zip	InLineZip.dll	Performs data compression
Manual Command	ManualCommand.dll	Generate a buffer containing user specific data and send this data onto another module
Null Transceiver	NullTransceiver.dll	Provides data display and muxing capabilities
Packet Processor	PacketProcessor.dll	Processes and displays statistics on CCSDS packetized telemetry

Packet Processor Version 2	PacketProcessorVer2.dll	Reassembles packets from input CCSDS version 1 and version 2 frames
Randomizer	Randomizer.dll	Provides CCSDS-defined randomization or de-randomization of input data unit
Raw Sockets	RawSockets.dll	Provides bi-directional, connectionless socket
Soft Synchronizer	SoftSync.dll	Bit synchronizes serial telemetry streams
Stripper	Stripper.dll	Allows user to modify a data stream by moving any size pieces of data to different offsets within the buffer, or eliminating them entirely
TDM Simulator (TDMSim)	Tdmsim.dll	Provides user-configurable source of TDM telemetry
Time Delay	TimeDelayTest.dll	Performs full loop delay/jitter testing
Unknown Module	UnknownModule.dll	Loaded automatically when a module exists on the Server but not on the machine running Console
Encapsulation/Extraction Modules		
Name	**DLL**	**Description**
ACE SFDU Formatter	AceSfduFormatter.dll	Outputs an ACE SFDU formatted data stream
Ames Command Encapsulator	Ames CommandEncapsulator.dll	Adds configurable fields to commands in preparation for transmission
AXAF SFDU Formatter	AxafSfduFormatter.dll	Outputs an AXAF SFDU formatted data stream
DDD Formatter	DDDFormatter.dll	Adds and/or removes encapsulation formats to/from NASCOM Blocks per user-configurable setup. Supported formats include JPL-defined DDD header
DST SFDU Formatter	DstSfduFormatter.dll	Outputs a DST SFDU formatted data stream
EDOS Service Header	ESHFormatter.dll	Prepends EDOS Service Header to a buffer
IPDU Formatter	IPDUFormatter.dll	Prepends an IPDU header to a buffer
IPDU Receiver	IPDUReceiver.dll	Receives IPDU formatted packets from the network
ITCU Command	ITCUCommand.dll	Provides socket connection for ITCU command input and throughputs commands with ITCU header stripped
ITCU Telemetry	ITCUTelemetry.dll	Outputs data with a ITCU header appended to a socket
LEO-T CDH Formatter	LeoT CDHFormatter.dll	Prepends LEO-T Command Delivery Header to a command buffer
LEO-T CDH Receiver	LeoT CDHReceiver.dll	Receives LEO-T formatted commands
LEO-T TFDH Transmitter	LeoT TFDHTransmitter.dll	Prepends the LEO-T Telemetry Frame Delivery Header to a telemetry buffer
LEO-T TFDH Receiver	LeoT_TDFHReceiver.dll	Accepts frames with LEO-T TFDH appended to each from a socket. Optionally strips and outputs data
NASCOM Blocker	NascomBlocker.dll	Outputs NASCOM 4800 Bit Blocks

NASCOM Deblocker	NascomDeblocker.dll	Receives NASCOM 4800 Bit Blocks from the network and outputs a serial data stream
NASCOM RTP Formatter	NascomRTPFormatter.dll	Receives a 4800BB and outputs an IP encapsulated frame
NASCOM RTP Receiver	NascomRTPReceiver.dll	Receives an IP encapsulated frame and outputs a NASCOM 4800BB.
SFDU 3-29 Formatter	Sfdu3_29Formatter.dll	Formats input data, appends TLM-3-29 SFDU Header to each input buffer, and outputs the SFDU.

CREATING A DESKTOP

The steps to create a new desktop from scratch follow:

1) Connect the GUI: Connect the Console in control mode. Use *System Menu/Connect Control To* option.
2) Clear the Console: Issue a *Remote Desktop Menu/New Desktop* command, which clears all module information from the Console. A blank Console window will appear.
3) Add the Required Modules: Use the *Add Module* function in the *Remote Desktop* menu to load each module that is required for the desktop. Modules are loaded one at a time.
4) Make Data and Event Connections: Carefully make the data and event connections that define each module's I/O. Use the *Remote Desktop Menu/Edit Module Connections* option.
5) Configure the Modules: Configure each of the modules. Configuration is done via each module's window that appears on the Console desktop.
6) Save the Desktop: Save the desktop configuration using the *Remote Desktop Menu/Save Desktop* option.

NOTE: A *New Desktop* command clears all modules and connections currently loaded in the window. As a result, if the user is working with an unsaved desktop when the *New Desktop* option is issued, then the current unsaved module information is cleared and lost.

The steps to create a new desktop based on an existing desktop follow:

1) Connect the GUI: Connect the Console in control mode. Use *System Menu/Connect Control To* option.
2) Clear the Console: Issue a *Remote Desktop Menu/New Desktop* command, which clears all module information from the Console. A blank Console window will appear.
3) Load the Desktop: Load a previously saved desktop by name. To load a desktop from the local machine, use *Remote Desktop Menu/Load Desktop*. To load a desktop from a remote machine, use *Remote Desktop Menu/Load Desktop From Network.*
4) Rename and Save the Desktop: Rename and save the desktop before making any changes. Use *Remote Desktop Menu/Save Desktop As.* Do NOT use the *Save Desktop* option,

which saves the current changes without prompting and does not allow the desktop to be renamed.

5) Add and Remove Modules: To remove a module, highlight the module and issue a *Remove Module* from *Remote Module* pull-down menu. To add a module, use *Remote Desktop Menu/Add Module.*

6) Edit Connections and Configuration: Edit connections and events between all modules. Edit each module's configuration to reflect the new setup. Use the *Remote Module Menu/Edit Module Connections* option for each highlighted module.

7) Reconfigure the Modules: If necessary, reconfigure each of the modules. Configuration is done via each module's window that appears on the Console desktop.

8) Save the Desktop: Issue a *Remote Desktop Menu/Save Desktop* when all changes and additions are complete.

Adding a Module

The desktop data flow is defined by loading and connecting modules. The procedure to add a module *(also referred to as load module)* follows:

1) From the *Remote Desktop* pull-down menu, select *Add Module*.

2) The *Add Module* window (**Figure 3-12**) will appear, with all access windows activated.

3) From the pull-down list, select the module to be opened (<filename>.DLL).

4) Select either *Load* or *Add Module.* Using *Load* adds the module to the desktop and closes the *Add Module* window. Using *Add Module* adds the module to the desktop, but keeps the window open so additional modules can be added without requiring that the *Add Module* window be re-opened.

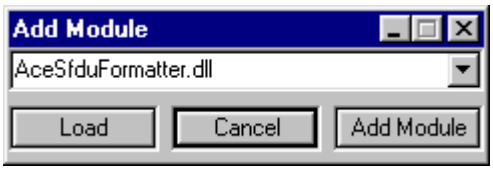


Figure 3-12: Add Module Dialog Window

Figure 3-13 provides an example of a Console Window with four Modules loaded.

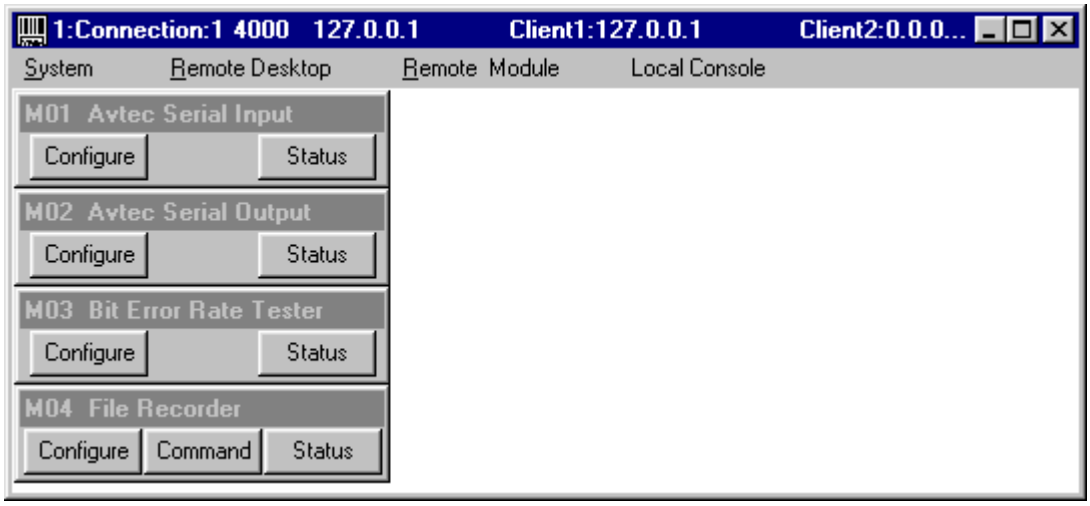


Figure 3-13: Console Window with Modules Loaded

Connecting Module Data and Events

Data I/O Modules and Data Processing Modules are "connected" together to perform real-time data processing tasks. As part of a Desktop, the user defines the data flow between the various modules. Each Data Module has data I/O, and may also have event I/O.

To edit module connections, select *Remote Module Menu/Edit Module Connections*. An *Edit Module Connections* window will appear, as shown in **Figure 3-14**. Using the *Edit Module Connections* from the *Remote Module* menu will bring up the dialog box on the highlighted Module. Using the *Edit Module Connections* from the *Remote Desktop* menu will bring up the dialog box on the first Module (M01), whether or not another Module is highlighted.

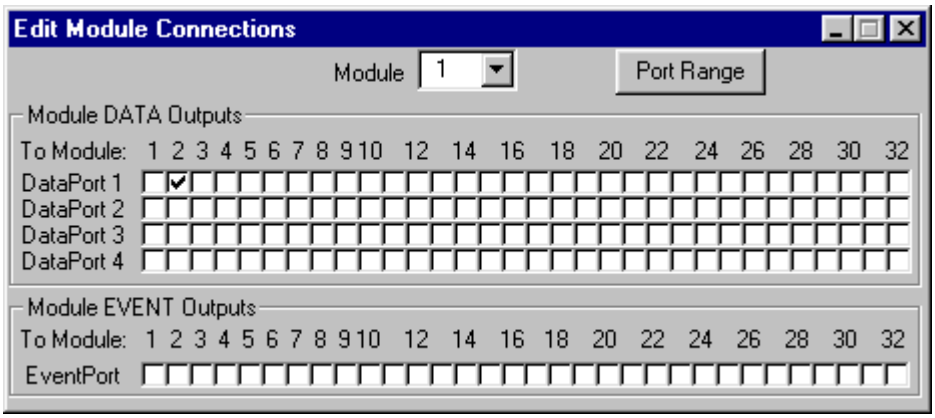


Figure 3-14: Edit Module Connections Window

The *Edit Module Connections* window allows every Module that has been added to a desktop to be configured for data and event I/O. It is the operator's responsibility to know what outputs and ports are applicable to each Module.

In the window, the Module number that all information on the window pertains to is defined in the center-top. To change the Module number and access connections to a different

Module, the operator uses the down-arrow button, which lists all other module numbers that are part of the desktop, and then highlights the desired module number.

The input of a Module is defined by the output configuration of the source Module. The Module output destinations are defined by the check marks that are entered into the window. The source module is defined in window top-center. The destination module numbers are listed horizontally across the screen. (Source module numbers are fixed display values for this screen and they do not indicate that a module with that number has actually been added to the desktop.) A check beneath a destination module number indicates that the output on that port will be input to that module number. Output on one port can go to numerous destinations.

On the window, each Module can be setup with four possible output ports and one possible event output. Clicking on the "Port Range" button will cycle through the next four DataPort numbers. When DataPort 32 is reached, it will cycle back to DataPort 1. Like the destination module numbers, these are fixed display values that are part of the window display. Not all Modules use all four available output ports and not all modules have an event output. Many modules also do not accept an event input. One or two modules also have additional output ports (more than four) that can be accessed via RIL calls. Refer to **Chapter 4** to see which output ports apply to each module.

In general, a Module outputs its "standard" data output to Port one. Many modules have no other active output ports.

Data Connections

A module's data output is connected to another module's input to pass data buffers from the first module to the second.

Event Connections

A module's event output is connected to another module's input to signal an event to the second module. For example, the Serial Output Module provides an event output every time it completes writing a frame of data to the I/O board. To playback a binary file, the Playback Module's data output is input to the Serial Output Module, and the Serial Output Module's event output is input to the Playback Module. The event connection provides a flow control between the two modules.

SAVING AND LOADING A DESKTOP

Once a Desktop has been created or modified, it can be saved for future use. Typically, the Avtec PTP for Windows has several different desktops saved, and each is used for a different application.

Saving a Desktop

To preserve configuration information, a Desktop needs to be saved before a *New Desktop* command is issued. At a save command, the configuration of each Module at the last *SEND* issued is the configuration that is saved. (The *SEND* command must be used correctly in all

Module configuration windows in order for the module to be set up as expected. **Chapter 4** discusses the SEND command in greater details.)

Avtec PTP for Windows either saves the desktop under its current name (*Save Desktop*) or it allows the desktop to be renamed (*Save Desktop As*).

To save a Desktop, proceed as follows:

1) From the *Remote Desktop* pull down menu, select *Save Desktop As*.
2) The Save Desktop window (**Figure 3-15**) will appear, with all access windows activated.
3) Type a file name in the window. We suggest you maintain the naming convention of .DTP (<filename>.DTP) for the file extensions.
4) Click *Save*.

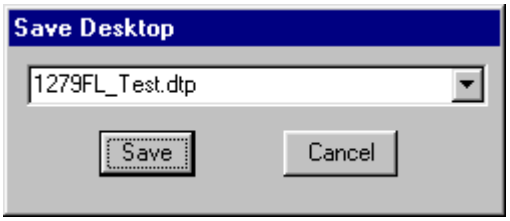


Figure 3-15: Save Desktop Window

Save Desktop As also allows you to rename the current desktop as a new file, thereby preserving the original as well.

Save Desktop saves the current configuration setting to the currently defined file name. **Note: No warning is given before overwriting a current file.** *Save Desktop to Network* allows you to save the current desktop to a folder located anywhere on the local machine or LAN.

Loading a Saved Desktop

The *Load Desktop* option allows you to load previously stored desktops, as shown in **Figure 3-16**. It lists all the desktops saved in the ptp_user\desktops directory. The operator selects the required desktop (the list is accessed via the down arrow), and then loads it onto the Console. The window provides two load buttons and either can be used to load the desktop. If the "Load" button is used, then the desktop is loaded and the dialog window is closed. If the "Load Desktop" button is used, then the desktop is loaded, but the dialog window remains open.

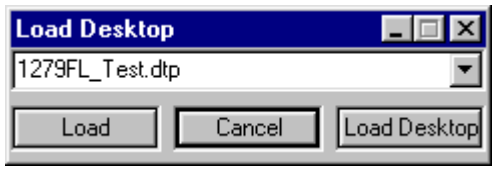


Figure 3-16: Load Desktop Window

NOTE: If a desktop was saved with the modules enabled, then the desktop comes up in an enabled state at Load, and desktop modules become active immediately.

The *Load Desktop from Network* option allows you to load a desktop from any folder on the local machine or from a LAN. In order to avoid the –200 Error, the user must have the necessary rights and privileges to access the LAN, and the folder where the desktops are stored must be shared. If a drive on a remote machine is shared, its shared name must be the drive letter (e.g. C or D) and not any other fancy name.

1) From the Remote Desktop pull down menu, select Load Desktop from Network.
2) A window shown in **Figure 3-17** will appear. The directory of all desktops is ptp_user\desktops. Select the desktop to be opened (<filename>.DTP).
3) Click Open.

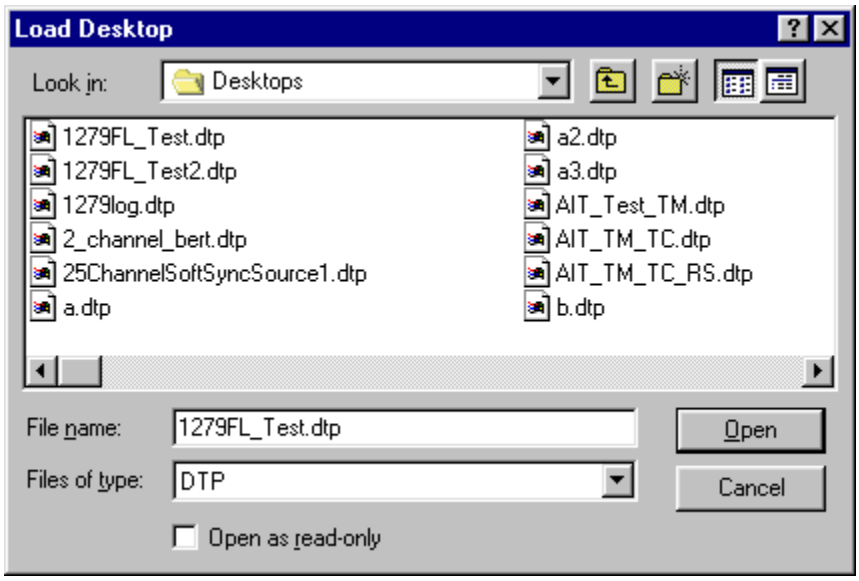


Figure 3-17: Load Desktop From Network Window

DESKTOP CONTROL FUNCTIONS/SYSTEM ENABLE AND DISABLE

To control data flow, use the following commands, located in the *Remote Desktop* main pull down menu:

- *Enable All Streams* – Allows you to enable all of the modules on your desktop at once. Selecting this command disables all streams and then enables them all again.
- *Disable All Streams* – Allows you to disable all modules. To restart data flows, select *Enable All Streams*.
- *Zero All Counts* – Returns all counters on the desktop to zero.
- *Set Status Delay* – Sets the delay time between multicast module status messages in milliseconds (minimum is 100, default is 1000). To set status delay:

  1) From the *Remote Desktop* pull down menu, select *Set Status Delay*.

2) A *Dialog* window (**Figure 3-18**) will appear.
3) Enter the desired delay time.
4) Click *Set*.

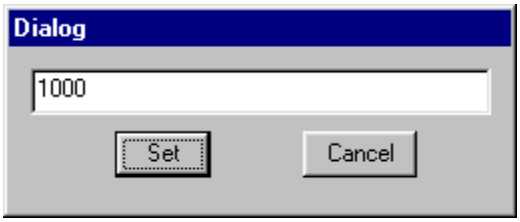


Figure 3-18: Dialog Window

- *Set Activity Time Out* – Sets the maximum time between module inputs/outputs before the activity monitor will indicate no data. To set activity timeout, select *Set Activity Time Out* and follow the steps above. Default is 1000 msec.
- *Set Station Id* – Allows you to specify the Station ID for the Server. To set the Station ID, select *Set Station ID* and follow the steps above.
- *Set Status TTL* – Allows the user to set the Time-to-Live (TTL) for status messages, e.g. the number of hops a status message takes before it is killed. To set the status message TTL, select *Set Status TTL* and follow the steps above.
- *Secure Buffer Mode (Enable/Disable)* – This command enables or disables *Secure Buffer Mode* operation of the Avtec PTP for Windows. When data is passed from one module to another, what is actually passed is a "pointer" to the data. If the same data is passed to multiple modules, all modules that are to receive the data will get the same pointer. This can be a problem if any one of these modules is going to alter the data in any way. With *Secure Buffer Mode* enabled, when one data stream is passed to multiple modules, a copy of the data is made for each module. In this case, if one module changes the data, it will not affect the data being used by the other modules. To enable or disable *Secure Buffer Mode*, select *Secure Buffer Mode* and then click on either *Enable* or *Disable*.

SECTION 5
SERVER HIGHLIGHTS

The Server is the real-time portion of the Avtec PTP for Windows software. When modules are being loaded and module configurations and connections are being made from the Console window, what is actually happening is that the Console is sending commands to the Server, and the Server is loading the modules and making the connections between them. The Console window is a GUI that simplifies the task of configuring the Avtec PTP for Windows and allows these tasks to be accomplished from a remote computer.

However, many of these commands can be entered as command strings from the Server window itself. Most of the functions available in the GUI have a command line counterpart in the Server window. To use the commands, the Server window must be made active and the command string is typed in as a one-line command similar to a command typed from a DOS Shell prompt. **Appendix A** contains a list of the Server commands and their respective definitions and usage.

AUTOMATED SETUP AND CONTROL (SCRIPT FILES)

The most useful feature of having the Server commands available is the ability to write script files to quickly load and configure various desktops. A script file is a text file containing Server Commands that are executed line-by-line by the Avtec PTP for Windows when the script file is run. A script file can be used to load modules, configure data and event connections, and enable modules. Much of the configuration of a module must still be completed via the Console GUI, however, a script file can be a quick way to get a desktop started.

CREATING SCRIPT FILES

This next example will use the Server commands in writing a script file to setup and configure connections in a BERT Loopback desktop.

To create a script file, open up a text editor such as the *Windows Notepad*. The BERT Loopback desktop can be built with the following string of commands:

```
NEWDESKTOP
ADDMODULE C:\PTP_NT\modules\M0202_AvtecSerialOutput.dll
ADDMODULE C:\PTP_NT\modules\M0201_AvtecSerialInput.dll
ADDMODULE C:\PTP_NT\modules\M1001_BitErrorRateTester.dll
ADDMODULE C:\PTP_NT\modules\M1000_NullTransciever.dll
ADDEVENT  1 3 0 0       /* Output Module Connection */
ADDOUTPUT 2 3 1 0       /* Input Module Connection */
ADDOUTPUT 2 4 1 0       /* Input Module Connection */
ADDOUTPUT 3 1 1 0       /* BERT Module Connection */
```

Save the file as *C:\ptp_user\scripts\BERTSCRIPT.PTP*. Open the Server and execute the following command:

READSCRIPTFILE C:\ptp_user\scripts\bertscript.ptp

After the file has been executed, the Server display should have four modules loaded, and the status for module 4 (the Null Transceiver Module) should be the active display. If further configuration of the modules is needed, then a Console window needs to be connected to the Server, and the configuration changes have to be made via the GUI.

The remaining desktops that were described in **Section 4** of this Chapter can be loaded in a similar fashion. A good exercise in writing script files would be to recreate those desktops using the Server Commands. This will be left to the reader.

SECURING CONTROL SOCKETS AND TCP SERVER SOCKETS

All TCP Server mode connections are socketed connections by definition. All modules that use network connections have an entry for "PTP Port" that controls the local socket number. TCP Server mode connections are not limited to modules, but also include Server Control Ports (sockets 4000 and 5000). A Well-Known Hosts file can be created for each local socket number, limiting access to that particular port to a select set of users.

When a client attempts to connect to a TCP Server mode socket (port) on an Avtec PTP for Windows machine, the PTPAPI sockets library will check for a Well-Known Hosts file. A Well-Known Hosts file is created by a Avtec PTP for Windows administrator, and contains a list of all clients that are allowed to connect to a particular socket. If the file does not exist, all clients will be allowed to connect, either to control the Server or send/receive data to/from the Server. If the file exists, the Avtec PTP for Windows compares the client's IP address and/or subnet mask to those listed within the Well-Known Hosts file. If the client is listed within the file, the connection is allowed, otherwise, it is refused. If the file exists, but is empty, no clients will be allowed to connect. In either case where the socket connection is refused, the Avtec PTP for Windows simply disconnects the socket and recycles for the next client.

The Well-Known Hosts file is a text file that is stored locally on an Avtec PTP for Windows machine. The naming convention is as follows:

\ptp_user\security\Sxxxx.txt

The directory ***"\ptp_user\security"*** is located on the same local drive as the Avtec PTP for Windows application. ***"xxxx"*** in the filename represents the socket (port) number in decimal.

The file can be created using a simple text editor, such as Windows Notepad. The file will be similar to the one shown below. It will contain a list of all clients allowed on a particular socket, in standard dotted notation. Only clients shown on this list will be allowed to connect to a particular socket. As shown below, subnet masks are allowed by following the client's IP address with a "#" and the subnet mask.

```
192.100.100.15
200.200.200.6
198.192.200.110 #255.255.255.0
206.184.110.100 #255.255.255.0
```

Chapter 4
Configuration and Status

SECTION 1
MODULES OVERVIEW

The basic operational unit in the Avtec PTP for Windows software is a module. A module is a separately maintained run-time link program that performs user-specific processing in real time. Modules are the elements that work together to form an Avtec PTP for Windows desktop. Each module has unique capabilities and settings.

MODULE WINDOW

For each Module added to the Console Desktop, a module window appears. **Figure 4-1** shows an example module window. The Title Bar for each module window contains the module number *(e.g., M01)* and the module name *(e.g., File Recorder).* Module windows may contain up to three buttons to control and monitor the module (all three buttons do NOT appear on every module window): *Configure, Command,* and *Status*.

Figure 4-1: Example Console Module Window

Configure Button

The *Configure* button brings up the configuration window that contains all configurable parameters for that Module. It is important to note that a configuration window may include an *Extended Configuration* button that accesses additional configuration information.

Each configuration window includes a *SEND* button that causes the values currently entered on the window to be set. Correct use of the *SEND* button is critical for setting up each module as expected. No configuration changes made to the window go into effect until the operator issues the *SEND* command. The *SEND* button is always on the top-level configuration window (the first window that opens from the module window), but it also affects the *Extended Options* window. Before closing the *Extended Options* window, the operator MUST issued the *SEND* command on the main configuration window or the *Extended Options* changes are lost. (The *SEND* command is not a Desktop save command).

A checkbox titled "Updates" exists on the configuration window that reports if a *SEND* has been issued since any parameter has been changed on the window. If *Updates* is checked, then no information on the configuration window has been changed since the last *SEND.*

To change any of the settings for any module, perform the following steps:

1) Click the module window *Configure* button to display the configuration dialog
2) Make changes to the Parameters (the Updates check box will automatically clear)
3) Click the *SEND* button to apply the new parameters *(the update box will automatically be re-enabled)*

Command Button

The *Command* button displays the Module command dialog window. The command dialog is used to send commands to the Module.

Most Modules do not have associated commands. For those that do, the commands can be issued real-time to control some feature of the Module data processing.

Status Button

The *Status* button displays the module status dialog. The status dialog displays the status of the module. Depending on the module, the status may display some static information that reflects the module setup. Almost every module's status includes real-time processing statistics that begin to increment when data flow through the module starts. Some status windows include an *Extended Status* button. If so, then more detailed processing information is available through button execution.

The Console continuously receives configuration and status information from the Server multicast output.

THE "SEND" BUTTON

The *SEND* command is part of the module configuration window. It is highlighted again in a separate section because it is an operational procedure that is very easily over-looked and misused by a first time operator, but it is also critical that it is used correctly.

Whenever a module configuration window is edited, the changes entered in the GUI are not set until the *SEND* command is issued. If an *Extended Options* configuration window is opened, the *SEND* must be issued from the mother configuration window before the *Extended Options* window is closed, or any edits done in the *Extended Options* window are lost.

The *SEND* command sets the changes made to that module's configuration and affects any data flow through the Module after the *SEND* is issued. However, remember that the configuration is not saved so that it can be recalled for use until the Desktop itself is saved.

SECTION 2
DETAILED MODULE DESCRIPTIONS

This section provides an alphabetical listing and description of all the Modules that can be added from the *Remote Desktop/Add Module* menu. For grouping by function type (Data I/O, Auxiliary I/O, Data Processing, and Encapsulation/Extraction) refer to **Table 3-7** in **Chapter 3**.

Module 1 8 BIT SORTER MODULE

MODULE NAME: Sorter8bit.dll

Key Functions

- Routes input data to different ports based on user defined data content
- Creates files containing the input data sorted on user defined content

Required Hardware

- None

Add Module Guidelines

There are no module iteration restrictions. One module can sort and switch on data content to supply up to four different user defined routes to other modules.

Required Input Source, Connections, and Events

INPUT SOURCE: All input data is passed to this module from other modules loaded on the desktop.

CONNECTIONS: Connections from at least one other module is necessary to supply data to this module. One or more output ports of this module must be connected to other modules to pass on the sorted / switched data.

EVENTS: No events are expected or generated.

Module Functional Description

The Sorter Module receives one or more streams from its input. Based on an 8 bit or smaller value in a user-specified byte location of each frame or block, the 8 bit Sorter will range test up to four user-specified ranges of values and send the frame or block out of the corresponding output port of the module. The 8 bit value that is picked out of the data may be bit-order reversed (least significant bit first), one's complemented (inverted), shifted, and masked as needed. Using the Avtec PTP for Windows "Edit Module Connections"

capability, each port of this module can be connected to the input of any other module. In fact, there does not need to be a one-to-one correspondence between output port number and another modules' input; two or more of the four ports can be connected to the same or different modules' input. No modifications or additions are made to the data content or time stamp of the buffer; this module only routes the data.

The frames or blocks (buffers) can also be recorded to separate disk files named as specified by the user, with a decimal value extension. The user should consider masking out extra bits of the selected byte when a range of data is being sent to this module. A separate file is opened, and pointers, buffers, and high priority execution threads are allocated in memory for each different value that results from the selected bytes after any bit-reversal, complementation, shifting, and masking is complete. Selecting a counter with a wide range will result in Windows NT attempting to maintain many (up to 256) open files. Masking off bits, either the high-order, low-order or some of each, will decrease the number of open files and generally improve system thruput and performance. To further sort or switch on data content, the data could be replayed post-pass from the sorted files, and further sorted, using different range masks. It is possible to overload the system, based on the amount of memory, data rate, bus speeds, disk speed and cache, and processor speed available. Degradation of lower priority threads, such as keyboard control or window updates is possible when the system becomes and remains overloaded; a process monitoring thread in the Avtec PTP for Windows application is designed to disable all processing, restoring control of the system once again.

Input Data Format

The only byte that is examined is the byte number specified by the user. This byte number is relative to one; the most-significant (first) byte in the buffer is numbered 1. Bytes are numbered sequentially through the last byte used. The content of all other bytes is ignored.

Output Data Format

The output data format is the same as the input format; this module only routes data.

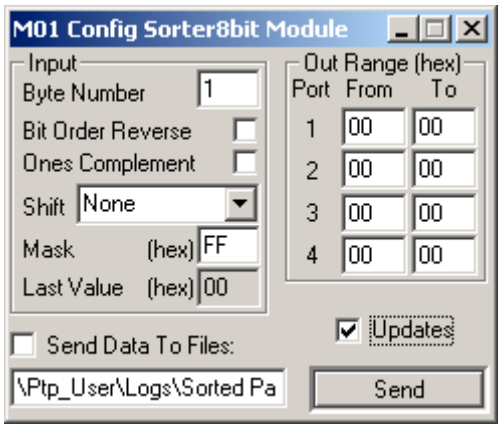
Module Window

When Sorter8bit.dll is added to a desktop, the 8 Bit Sorter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 1-1: 8 BIT SORTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears as shown below. The table that follows defines the fields in the Configure 8 BIT SORTER MODULE Board Parameters sub-menu.



Module 1-2: Config 8 BIT SORTER MODULE Window

Module 1-3: 8 BIT SORTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			
Byte Number	Decimal	1 to 16384	Byte number to sort / switch on; the first byte in the buffer is numbered 1. Care must be taken to sort on a byte number that is less than or equal to the actual length of the buffer. No software test exists for this; whatever byte number that is entered will be tested, even if that byte is located physically after the buffer content. This allows testing of data time and status that may be located in the buffer after the data.
Bit Order Reverse	Checkbox	Enable Disable	Bit Order Reversing. If selected, the eight bits of the byte selected will be re-ordered. For example, **10110001** would become **10001101**. The buffer content is unchanged.
Ones Complement	Checkbox	Enable Disable	Ones Complement Selected byte. If selected, the eight bits of the byte selected will be one's complemented (inverted). For example. **11010010** would become **00101101**. The buffer content is unchanged.
Shift	Selectable Setup	None, Left 1 to 7 bits, Right 1 to 7 bits	Shift data left or right, 1 to 7 bits. Bits shifted out of either end are lost. Bits shifted into either end (to fill vacant bit positions) will always be '0' bits. The buffer content is unchanged.
Mask	Hexadecimal	00 to FF	Mask pattern. This pattern is masked (ANDed) with the selected input byte. Bits set to '1' in the mask remain unchanged in the result. Bits set to '0' in the mask will be set to '0' in the result.
Last Value	Display only	00 to FF	Last value (in hex) from the input buffer.

Parameter	Unit	Range or Values	Description
OUT RANGE (HEX)			
Port 1 to 4	Hexadecimal	00 to FF	For each output port, define the range of values to be output on that port. The ranges are searched sequentially; the first range satisfied causes the data to be output on that corresponding port, and no further range checks take place. No buffer will be output on more than one port. If no range is satisfied, the buffer will not be routed to any port.
OTHER SETUP			
Send Data to Files	Checkbox	Enable Disable	Sorted data can be logged to disk directly from the Sorter module. Check the checkbox to enable recording to disk. The file name is user defined, the file extension is automatically set as the value of the 8 bit or smaller field in the data. The default filename is \Ptp_User\Logs\Sorted Packet.Rec. The value of the selected byte (after any bit-reversal, complementing, shifting, or masking) is appended to the user-specified file name as a dot followed by the decimal value of the selected routing byte. For example, a filename of \Ptp_User\Logs\Today.rec and a received value of decimal 37 would result in an output filename of \Ptp_User\Logs\Today.rec.37.

Extended Options Window Definition

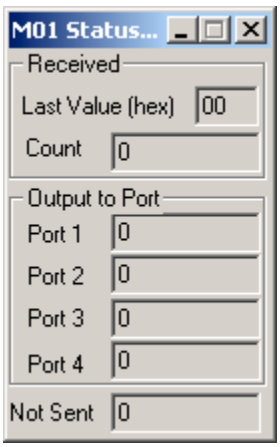
There is no extended options window for the 8 BIT SORTER MODULE.

Configuration Notes

There are no configuration notes for the 8 BIT SORTER MODULE.

Status Window Definition

When you click the Status button on the 8 BIT SORTER MODULE window, a Status 8 BIT SORTER window appears as shown below.



Module 1-4: Status 8 BIT SORTER MODULE Window

Module 1-5: 8 BIT SORTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
RECEIVED			
Last Value	Hexadecimal	00 to FF	Value of last data received (selected byte in hex). This value is reported after any bit-reversal, complementing, shifting, or masking is done.
Count	Decimal		Total number of frames/blocks received by Sorter
OUTPUT TO PORT			
Port 1 to 4	Decimal		Total number of frames/blocks sent out each Data Port.
OTHER STATUS			
Not Sent	Decimal		Total number of frames not sent because none of the user-defined ranges were satisfied. These frames (buffered) not routed.

Module 2 16 BIT SORTER MODULE

MODULE NAME: Sorter16bit.dll

Key Functions

- Routes input data to different ports based on user defined data content
- Creates files containing the input data sorted on user defined content

Required Hardware

- None

Add Module Guidelines

There are no module iteration restrictions. One module can sort and switch on data content to supply up to four different user defined routes to other modules.

Required Input Source, Connections, and Events

INPUT SOURCE: All input data is passed to this module from other modules loaded on the desktop.

CONNECTIONS: Connections from at least one other module is necessary to supply data to this module. One or more output ports of this module must be connected to other modules to pass on the sorted / switched data.

EVENTS: No events are expected or generated.

Module Functional Description

The Sorter Module receives one or more streams from its input. Based on a 16 bit or smaller value in a user-specified two byte contiguous location of each frame or block, the 16 bit Sorter will range test up to four user-specified ranges of values and send the frame or block out of the corresponding output port of the module. The 16 bit value that is picked out of the data may be bit-order reversed (least significant bit first), one's complemented (inverted), shifted, and masked as needed. Using the Avtec PTP for Windows "Edit Module Connections" capability, each port of this module can be connected to the input of any other module. In fact, there does not need to be a one-to-one correspondence between output port number and another modules' input; two or more of the four ports can be connected to the same or different modules' input. No modifications or additions are made to the data content or time stamp of the buffer; this module only routes the data.

The frames or blocks (buffers) can also be recorded to separate disk files named as specified by the user, with a decimal value extension. The user should consider masking out extra bits of the selected byte when a range of data is being sent to this module. A separate file is

opened, and pointers, buffers, and high priority execution threads are allocated in memory for each different value that results from the selected bytes after any bit-reversal, complementation, shifting, and masking is complete. Selecting a counter with a wide range will result in Windows NT attempting to maintain many (up to 65536) open files. Masking off bits, either the high-order, low-order or some of each, will decrease the number of open files and generally improve system thruput and performance. To further sort or switch on data content, the data could be replayed post-pass from the sorted files, and further sorted, using different range masks. It is possible to overload the system, based on the amount of memory, data rate, bus speeds, disk speed and cache, and processor speed available. Degradation of lower priority threads, such as keyboard control or window updates is possible when the system becomes and remains overloaded; a process monitoring thread in the Avtec PTP for Windows application is designed to disable all processing, restoring control of the system once again.

Input Data Format

Only two bytes of the buffer are examined; the byte number specified by the user and the following byte. This byte number is relative to one; the most-significant (first) byte in the buffer is numbered 1. Bytes are numbered sequentially through the last byte used. The content of all other bytes is ignored.

Output Data Format

The output data format is the same as the input format; this module only routes data.

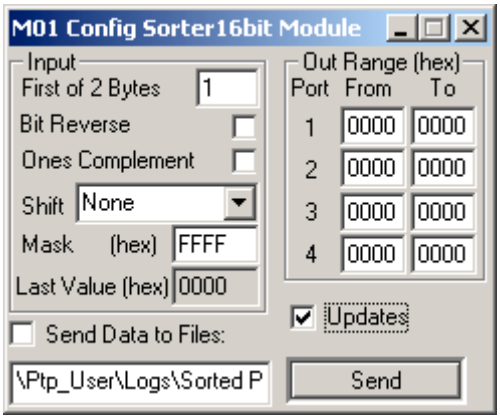
Module Window

When Sorter16bit.dll is added to a desktop, the 16 Bit Sorter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 2-1: 16 BIT SORTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure 16 BIT SORTER MODULE Board Parameters sub-menu.



Module 2-2: Config 16 BIT SORTER MODULE Window

Module 2-3: 16 BIT SORTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			
First of 2 Bytes	Decimal	1 to 16383	Byte number to sort / switch on (only the most significant byte is entered); the first byte in the buffer is numbered 1. Care must be taken to sort on a set of bytes that is less than or equal to the actual length of the buffer. No software test exists for this; whatever bytes are entered will be tested, even if that entry is located physically after the buffer content. This allows testing of data time and status that may be located in the buffer after the data.
Bit Reverse	Checkbox	Enable Disable	Bit Order Reversing. If selected, the sixteen bits of the entry selected will be re-ordered. For example, **1011000111110001** would become **1000111110001101**. The buffer content is unchanged.
Ones Complement	Checkbox	Enable Disable	Ones Complement Selected byte. If selected, the eight bits of the byte selected will be one's complemented (inverted). For example, **1000111111010010** would become **0111000000101101**. The buffer content is unchanged.
Shift	Selectable Setup	None, Left 1 to 7 bits, Right 1 to 7 bits	Shift data left or right, 1 to 7 bits. Bits shifted out of either end are lost. Bits shifted into either end (to fill vacant bit positions) will always be '0' bits. The buffer content is unchanged.
Mask	Hexadecimal	0000 to FFFF	Mask pattern. This pattern is masked (ANDed) with the selected input byte. Bits set to '1' in the mask remain unchanged in the result. Bits set to '0' in the mask will be set to '0' in the result.
Last Value	Hexadecimal	0000 to FFFF	Last value (in hex) from the input buffer.
OUT RANGE (HEX)			

Parameter	Unit	Range or Values	Description
Port 1 to 4	Hexadecimal	0000 to FFFF	For each output port, define the range of values to be output on that port. The ranges are searched sequentially; the first range satisfied causes the data to be output on that corresponding port, and no further range checks take place. No buffer will be output on more than one port. If no range is satisfied, the buffer will not be routed to any port.
OTHER SETUP			
Send Data to Files	Checkbox	Enable Disable	Sorted data can be logged to disk directly from the Sorter module. Check the checkbox to enable recording to disk. The file name is user defined, the file extension is automatically set as the value of the 16 bit or smaller field in the data. The default filename is \Ptp_User\Logs\Sorted Packet.Rec. The value of the selected byte (after any bit-reversal, complementing, shifting, or masking) is appended to the user-specified file name as a dot followed by the decimal value of the selected routing byte. For example, a filename of \Ptp_User\Logs\Today.rec and a received value of decimal 37 would result in an output filename of \Ptp_User\Logs\Today.rec.37.

Extended Options Window Definition

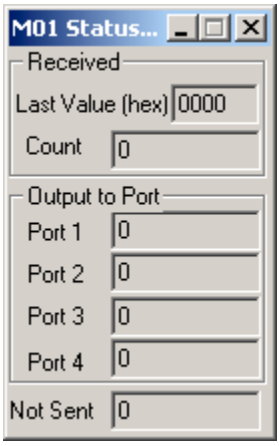
There is no extended options window for the 16 BIT SORTER MODULE.

Configuration Notes

There are no configuration notes for the 16 BIT SORTER MODULE.

Status Window Definition

When you click the Status button on the 16 BIT SORTER MODULE window, a Status 16 BIT SORTER window appears, as shown below.



Module 2-4: Status 16 BIT SORTER MODULE Window

Module 2-5: 16 BIT SORTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
RECEIVED			
Last Value	Hexadecimal	00 to FF	Value of last data received (selected bytes in hex). This value is reported after any bit-reversal, complementing, shifting, or masking is done.
Count	Decimal		Total number of frames/blocks received by Sorter
OUTPUT TO PORT			
Port 1 to 4	Decimal		Total number of frames/blocks sent out each Data Port.
OTHER STATUS			
Not Sent	Decimal		Total number of frames not sent because none of the user-defined ranges were satisfied. These frames (buffered) not routed.

Module 3 ACE SFDU Formatter MODULE

MODULE NAME: AceSfduFormatter.dll

Key Functions

- Encapsulates input frame with a 100-byte SFDU header
- Automatically fills fixed header fields
- Provides user-configurable header fields
- Automatically fills header fields that change with data input

Required Hardware

- The ACE SFDU FORMATTER MODULE does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

The number of ACE SFDU FORMATTER MODULE that can be added to a desktop is system dependent. A maximum of 32 modules can be added to any desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module may provide frame input to the module. However, typical use is data that has been processed by both the Serial Input and VC Processor modules so that status information is available to fill in the SFDU Header. The module can process input that does not include status information.

CONNECTIONS: Outputs data to port 1, which must be connected to the input of another module or the data is lost.

EVENTS: Neither generates an event nor expects an event input.

Module Functional Description

The ACE SFDU FORMATTER MODULE supports encapsulation of telemetry frames with a Standard Formatted Data Unit header. The header format is based on the definitions as needed for the Advanced Composition Explorer. Frame synchronization states are recorded within the SFDU header only if a serial input module using the MONARCH-E board is loaded as a source of the telemetry stream. In addition, the Reed Solomon decoder and Extended Status must be enabled.

Note: See DSN Document 820-13, Rev. A – TLM-3-27 *DSN Telemetry Interface with Advanced Composition Explorer (ACE)* for a detailed overview of the ACE SFDU format.

Input Data Format

Module expects CCSDS Version 1 or Version 2 frame input.

Any size frame can be input. Some information required for header generation becomes unavailable if the input frame does not include Avtec PTP for Windows status that was generated by the Serial Input Module and a VC Processor Module. Can be used as a wrapper for TDM input, but some status isn't applicable.

Output Data Format

Outputs in SFDU format: 100 byte SFDU header followed by telemetry data field that contains the frame input to the module.

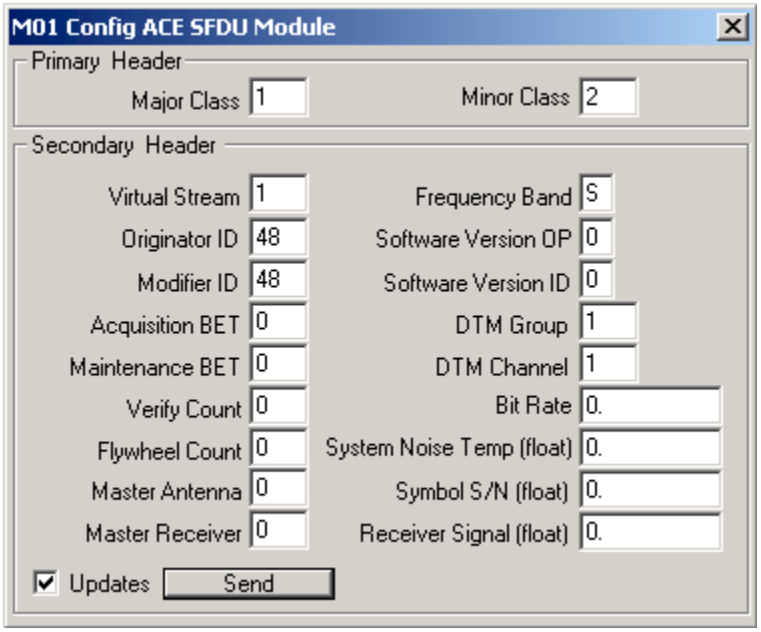
Module Window

When *AceSFDUFormatter.dll* is added to a desktop, the ACE SFDU Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 3-1: ACE SFDU FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure ACE SFDU FORMATTER MODULE Board Parameters sub-menu.



Module 3-2: Config ACE SFDU FORMATTER MODULE Window

Module 3-3: ACE SFDU FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
The ACE SFDU format is a 100 byte header that includes fixed fields, fields that reflect data input, and fields that are user-configurable. The setup fields provided in this window are only those fields that are defined by the operator. The range provided for each field is the number of bits assigned by the format definition. The description provides the header field name and some of the standard setup values used for that field. The operator can enter any value into the field that does not exceed the bit range. Default values show up when the window is brought up: for most fields the default value is zero.			
PRIMARY HEADER			
Major Class	decimal	8-bit field byte 29	Major Data Class; Default = 1 (Spacecraft telemetry data)
Minor Class	decimal	8-bit field byte 30	Minor Data Class. Indicates configuration of DSCC (Deep Space Communications Complex) frame sync and Reed-Solomon decoding processes. Valid values = 0, 1, 2
SECONDARY HEADER			
Virtual Stream	decimal	8-bit field byte 40	Virtual Stream Identifier; value = 1, 2, 64 (i.e. virtual channel 1)
Originator ID	decimal	8-bit field byte 37	Indicates message origin; 48=originated from DSN (Deep Space Network)
Modifier ID	decimal	8-bit field byte 38	Indicates where message was last modified; 48=last modified by DSN
Acquisition BET	decimal	8-bit field byte 55	Acquisition Bit Error Tolerance – defines number of allowable bit errors in sync pattern in search and verify modes; values=0 – 15

Parameter	Unit	Range or Values	Description
Maintenance BET	decimal	8-bit field byte 56	Defines number of allowable bit errors in sync pattern in lock and flywheel modes; values=0 – 15
Verify Count	decimal	8-bit field byte 57	Number of frames required for frame sync logic to transition from verify to lock state; values=0 – 15
Flywheel Count	decimal	8-bit field byte 58	Number of frames required for frame sync logic to transition from flywheel to search state; values=0 – 15
Master Antenna	decimal	8-bit field byte 87	Indicates which DSS antenna data was received from.
Master Receiver	decimal	8-bit field byte 88	Indicates which receiver was used with Master Antenna
Frequency Band	ascii	8-bit field byte 66	Indicates frequency band of data; values = S, X, K
Software Version OP	ascii	8-bit field byte 93	Software Version Operational Level; assigned by DSN Operations
Software Version ID	ascii	8-bit field byte 94	Software Version Identifier; assigned by DTM CDE
DTM Group	decimal	8-bit field byte 89	DSCC Telemetry Subsystem Group number
DTM Channel	decimal	8-bit field byte 90	DTM Channel number
Bit Rate	Bps float	32-bit field Refer to float format in DSN Document 820-13 bytes 67-70	Bit Rate in bits per second
System Noise Temp (float)	kelvin	32-bit field Refer to float format in DSN Document 820-13 bytes 73-76	System Noise Temperature in Kelvin
Symbol S/N (float)	dB	32-bit field Refer to float format in DSN Document 820-13 bytes 77-80	Estimated signal-to-noise ratio (dB) in symbol domain (prior to Viterbi decoding)

Parameter	Unit	Range or Values	Description
Receiver Signal (float)	dBm	32-bit field Refer to float format in DSN Document 820-13 bytes 81-84	Receiver signal level (dBm) for master receiver in array

Extended Options Window Definition

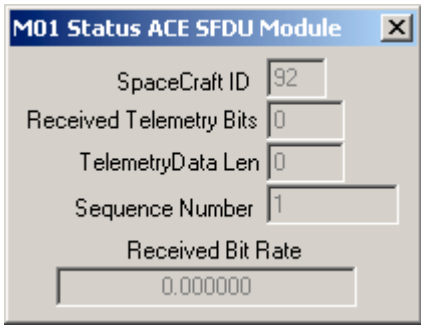
There is no extended options window for the ACE SFDU FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the ACE SFDU FORMATTER MODULE.

Status Window Definition

When you click the Status button on the ACE SFDU FORMATTER MODULE window, a Status ACE SFDU FORMATTER window appears, as shown below.



Module 3-4: Status ACE SFDU FORMATTER MODULE Window

Module 3-5: ACE SFDU FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
SpaceCraft ID	decimal	92	Hard coded to 92 to represent ACE spacecraft.
Received Telemetry Bits	bits	Detected value from frame input	Input data length in bits (buffer length) Reports the length of the telemetry data portion that follows the SFDU header
Telemetry Data Len	bytes	Detected value from frame input	Equals input data length (buffer length) Reports the length of the telemetry data portion that follows the SFDU header. This value will reflect the frame length input to the module.

Parameter	Unit	Range or Values	Description
Sequence Number	Integer SFDU	Incrementing counter	Count of the number of SFDUs output. Value is also put in SFDU header. Sequence number starts with one and increments for each SFDU output. Values are 1 to 2^32-1: after which the counter rolls over.

Module 4 AMES COMMAND ENCAPSULATOR MODULE

MODULE NAME: AmesCommandEncapsulator.dll

Key Functions

- Encapsulates command packets for serial output
- Can be used for generic insertion of sync marker
- Accepts packet input, wraps a header and trailer around each, and then outputs the packet

Required Hardware

- No special hardware required.

Add Module Guidelines

Do not exceed the limit of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide the data input.

CONNECTIONS: Outputs to port one

EVENTS: Does not require events. Can be used to generate a simulated telemetry flow in which case events are cascaded. For cascaded events, it receives an event input and modifies an event output to match the input minus the total number of bytes of the wrapper. Event rate stays the same, but the number of bytes requested is different.

Module Functional Description

The Ames Command Encapsulator Module is used to encapsulate command packets for serial output. The Command Encapsulator receives a command packet and prepends a Preamble and a BarkerCode and appends a Postamble. The Command Encapsulator outputs the new packet.

Receives in a buffer of data and then appends the wrapper info. First it generates a preamble, then the barker code, then the buffer of data, and then postamble

Input Data Format

Accepts buffer data input from any Module. Input does not have to be a fixed buffer size.

Output Data Format

Output is the input data with a wrapper appended. Output is one wrapper per buffer of data.

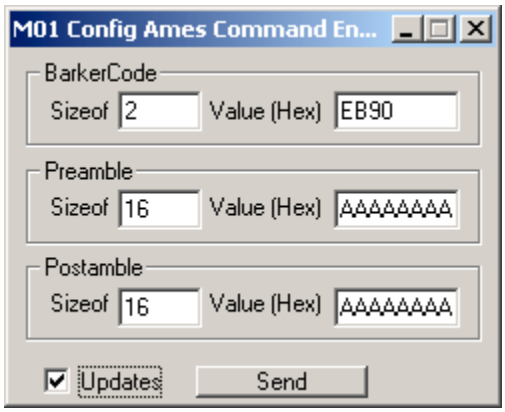
Module Window

When AmesCommandEncapsulator.dll is added to a desktop, the Ames Command Encapsulator dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click the Command button to access control information The status button is not implemented.

Module 4-1: AMES COMMAND ENCAPSULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AMES COMMAND ENCAPSULATOR MODULE Board Parameters sub-menu.



Module 4-2: Config AMES COMMAND ENCAPSULATOR MODULE Window

Module 4-3: AMES COMMAND ENCAPSULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
If you don't want one of the fields to be generated, set the sizeof field to zero If receives in bursty data, then it outputs bursty data			
BARKER CODE			
Sizeof	bytes		Size of the Barker Code in bytes
Value	Hex		Value of barker code
PREAMBLE			

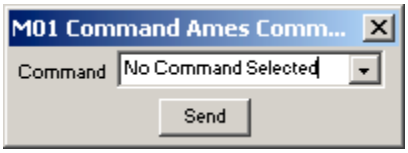
Parameter	Unit	Range or Values	Description
Sizeof	bytes		Size of the Preamble in bytes
Value	Hex		Value of Preamble
POSTAMBLE			
Sizeof	bytes		Size of the Postamble in bytes
Value	Hex		Value of Postamble

Extended Options Window Definition

There is no extended options window for the AMES COMMAND ENCAPSULATOR MODULE.

Configuration Notes

No commands are currently defined for the AMES COMMAND ENCAPSULATOR MODULE.



Module 4-4: Command AMES COMMAND ENCAPSULATOR MODULE Window

Module 4-5: AMES COMMAND ENCAPSULATOR MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	command	N/A	Makes the SEND command non-functional. No command to be issued.

Status Window Definition

There is no status window for the AMES COMMAND ENCAPSULATOR MODULE.

Module 5 APOGEE PSK MODULATOR MODULE

MODULE NAME: ApogeePsk.dll

Key Functions

- Sets up Apogee ISA-PSK BPSK Subcarrier Generator

Required Hardware

- The following board must be installed in the Avtec PTP for Windows system:
  - ✓ Apogee ISA-PSK BPSK Subcarrier Generator

Add Module Guidelines

One Apogee PSK Modulator Module can be added to the system for each device.

Required Input Source, Connections, and Events

INPUT SOURCE: Data input is external to device.

CONNECTIONS: The data output is external to device. No module connections are required.

EVENTS: This module neither generates an event or requires an event input.

Module Functional Description

The Apogee PSK Module (ApogeePSK) supports the Apogee ISA-PSK BPSK Subcarrier Generator. This module has no data inputs or outputs and cannot be "connected" to any other modules. All connections to and from this device are external, hardwired connections. This board accepts serial command data at various rates from an external source in NRZ-L format. This data is used to coherently modulate a 16 kHz subcarrier

Input Data Format

NRZ-L serial command data

Output Data Format

NRZ-L, NRZ-M, or NRZ-S format

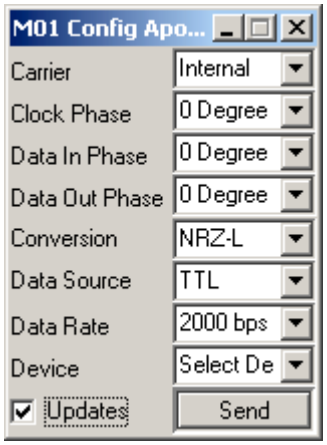
Module Window

When ApogeePsk.dll is added to a desktop, the Apogee PSK Modulator dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. This module does not have either a Command or Status window.

Module 5-1: APOGEE PSK MODULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure APOGEE PSK MODULATOR MODULE Board Parameters sub-menu.



Module 5-2: Config APOGEE PSK MODULATOR MODULE Window

Module 5-3: APOGEE PSK MODULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Carrier	Selectable Setup Parameter	Internal TTL RS-422	Selects subcarrier source
Clock Phase	Selectable Setup Parameter	0 Degree 180 Degree	0 or 180^0
Data In Phase	Selectable Setup Parameter	0 Degree 180 Degree	Input Data Phase
Data Out Phase	Selectable Setup Parameter	0 Degree 180 Degree	Output Data Phase

Parameter	Unit	Range or Values	Description
Conversion	Selectable Setup Parameter	NRZ-L NRZ-M NRZ-S	Code ConversionS
Data Source	Selectable Setup Parameter	TTL RS-422	Select Data Source
Data Rate	Selectable Setup Parameter	Off 125 bps 250 bps 500 bps 1000 bps 2000 bps	Select Data Rate in bps
Device	Selectable Setup Parameter	Available Devices	Selects which Apogee PSK Modulator is being configured

Extended Options Window Definition

There is no extended options window for the APOGEE PSK MODULATOR MODULE.

Configuration Notes

There are no configuration notes for the APOGEE PSK MODULATOR MODULE.

Status Window Definition

There is no status window for the APOGEE PSK MODULATOR MODULE.

Module 6 ATM MODULE

MODULE NAME: ATM.dll

Key Functions

- Sets up Forerunner LE ATM Adapter OC-3 board
- Maintains byte counts for data input and data output to/from module
- Allows user to define the ATM Cell VPI and VCI values that are stamped into the header of each ATM cell output from card.

Required Hardware

- Requires the following third party hardware installed in the Avtec PTP for Windows system:
  - ✓ Forerunner LE ATM Adapter OC-3 board (connects to fiber switch)

Add Module Guidelines

One module is required to control one board. Do not exceed the 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from any other module.

CONNECTIONS: Output is via the hardware.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

In conjunction with the Forerunner board, the ATM TRANSFER MODULE provides a bi-directional ATM OC-3 interface. It is message based delivery with no guaranteed delivery.

The module allows the user to define virtual channels: it allows the user to define VCI and VPI values that are the ATM Cell Header labels, which are used for routing that cell. Each ATM cell output is stamped with the defined VPI and VCI.

An ATM cell is 53 bytes: 48 byte cell with a 5 byte header.

Input Data Format

Buffer data input from any module.

Output Data Format

Output is 53-byte ATM cell: 48-byte data with 5-byte header.

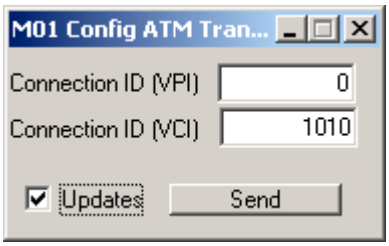
Module Window

When ATM.dll is added to a desktop, the ATM Transfer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 6-1: ATM TRANSFER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown in below. The table that follows defines the fields in the Configure ATM TRANSFER MODULE Board Parameters sub-menu.



Module 6-2: Config ATM TRANSFER MODULE Window

Module 6-3: ATM TRANSFER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Connection ID (VPI)		1-256	Defines cell header Virtual Path Identifier field. Stamps into the header for transmit, causes filtering if does not match at the firmware level
Connection ID (VCI)		1-65536	Defines cell header Virtual Channel Identifier field. Virtual channel on that hardware.

Extended Options Window Definition

There is no extended options window for the ATM TRANSFER MODULE.

Configuration Notes

There are no configuration notes for the ATM TRANSFER MODULE.

Status Window Definition

When you click the Status button on the ATM TRANSFER MODULE window, a Status ATM TRANSFER window appears, as shown in below.

Module 6-4: ATM TRANSFER MODULE Status Window

Module 6-5: ATM TRANSFER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing count	Count of user data input to module
Bytes Sent	bytes	Incrementing count	User data sent. Reports the data sent to the forerunner software

Module 7 AVTEC DEMOD BIT SYNC VITERBI MODULE

MODULE NAME: AvtecDemodBitSyncViterbi.dll

Key Functions

- Setup, Control, and Monitor Avtec's Demod/Bit Sync/Viterbi Board
- Provides user-configurable parameters for all board functions and reports status from the board

Required Hardware

- The following board must be installed in the Avtec PTP for Windows system:
  - ✓ Avtec Demod Bit Sync Viterbi (DBV) Board
- Appropriate cable to interface with the card.

Add Module Guidelines

Currently, up to eight boards can be controlled, using one module instance per board. The user specifies which board is to be controlled with the DEVICE parameter. The boards are numbered starting with 1, in the order that the cards appear electrically on the PCI bus. Some computers allow the user to specify the direction of the PCI bus scan (left to right or right to left) in the system BIOS.

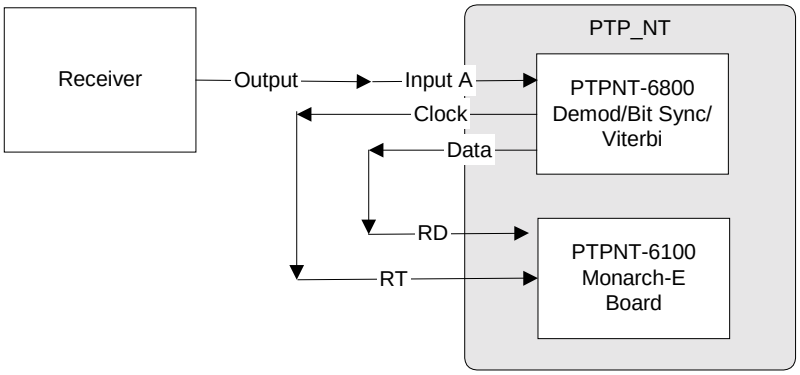
Required Input Source, Connections, and Events

INPUT SOURCE: All input is from physical cables attached to the board at the back of the computer.

CONNECTIONS: This module does supports an input connection and an output connection for time tag processing and quality annotation purposes. When a buffer of data is received from another module, the current DBV configuration parameters and status are copied into the buffer descriptor and the buffer is posted to the modules that are connected to the output port. This feature is used the TLM 3-29 SFDU module to set the header values for SNR and Telemetry Lock Status. The DBV also adjusts the time tag in the buffer descriptor to correct the time tag for the pipeline delay through the DBV board.

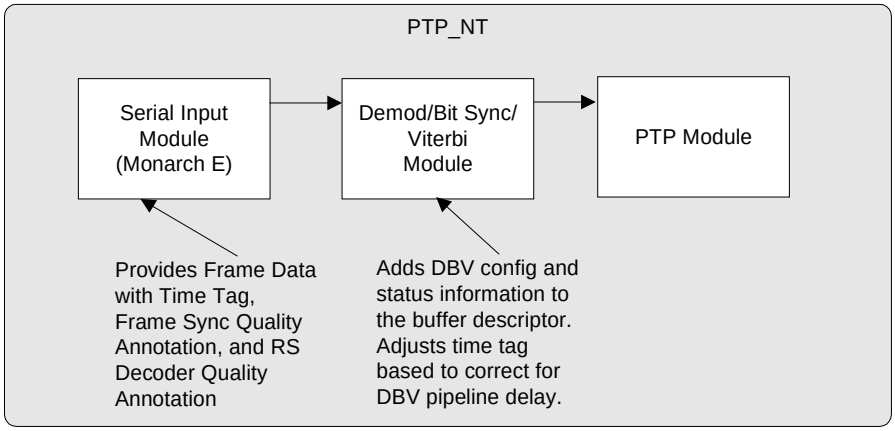
EVENTS: This module does not accept or produce any events.

The figure below shows the typical external hardware connections between a Receiver, the Demod/Bit Sync/Viterbi Decoder board, and a Monarch-E Frame Sync/RS Decoder board.



Module 7-1: External Hardware Connections

The figure below shows the typical module connections between a Serial Input module, a Demod/Bit Sync/Viterbi Module, and other modules, the Demod/Bit Sync/Viterbi Decoder board, and a Monarch-E Frame Sync/RS Decoder board.



Module 7-2: Module Connections

Module Functional Description

This module provides setup and control of the hardware card via the Console GUI or the Remote Interface Library (RIL). User selections for each parameter are passed to the driver, which controls the card via the PCI bus. Status values are obtained from the board and are available on the GUI or by RIL call.

Input Data Format

The user can specify the type of data that the hardware expects via control parameters.

Output Data Format

Output data format is specified by the user, either by the GUI or RIL call.

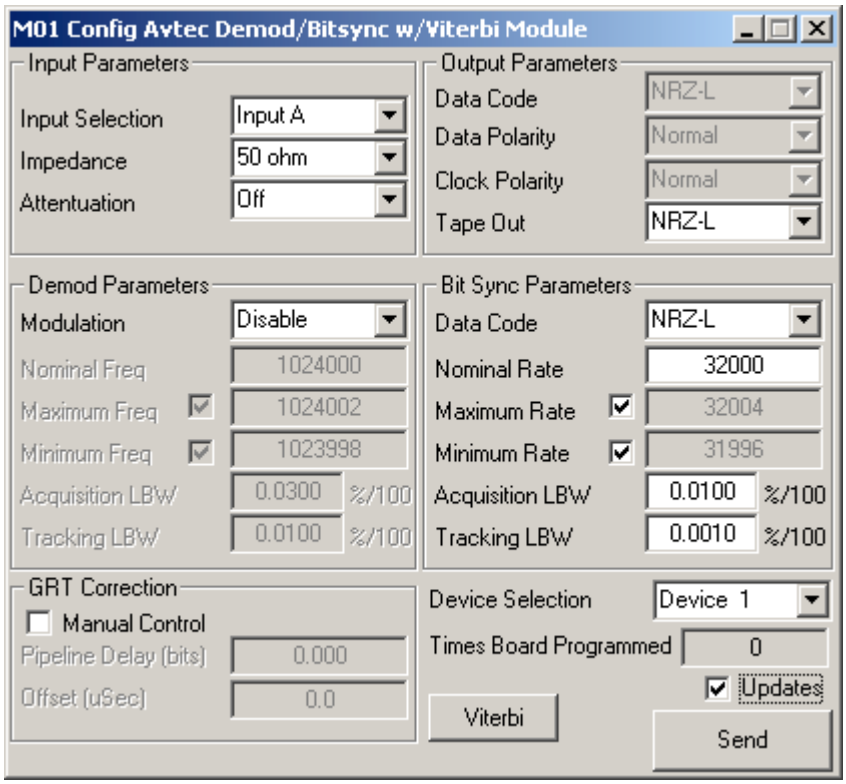
Module Window

When *AvtecDemodBitSyncViterbi.dll* is added to a desktop, the Avtec Demod Bit Sync Viterbi dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 7-3: AVTEC DEMOD BIT SYNC VITERBI MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure sub-menu.



Module 7-4: Config AVTEC DEMOD BIT SYNC VITERBI MODULE Window

Module 7-5: AVTEC DEMOD BIT SYNC VITERBI MODULE Configure Window Parameters

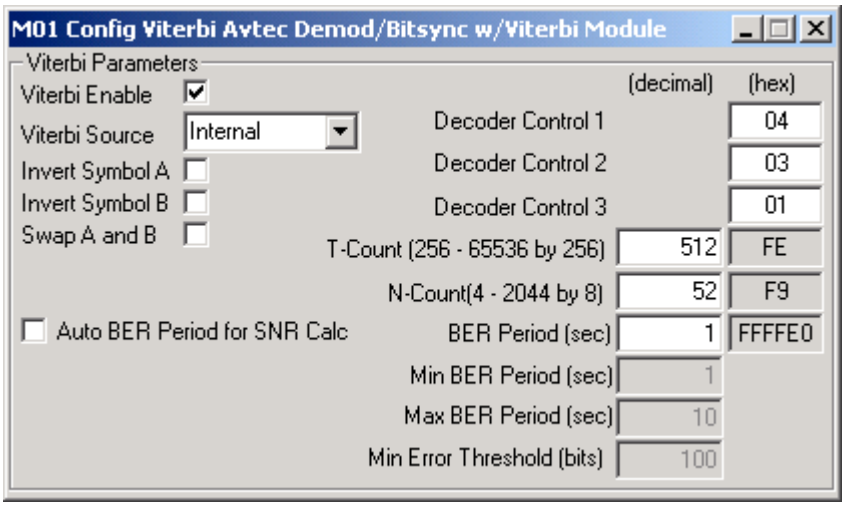
Parameter	Unit	Range or Values	Description
INPUT PARAMETERS			
Input Selection	Selectable Setup State	Input A, Input B, Differential	Selects which input (or differential) is to be processed by the board. Unselected signals are ignored.
Impedance	Selectable Setup State	10k ohm 50 ohm	Input impedance, low or high.
Attenuation	Selectable Setup State	Off On	Causes the board to attenuate the input by 5.2dbV (10.4dbW). $V_{attn} = 0.3 \times V_{in}$
OUTPUT PARAMETERS			
Data Code			Not implemented at this time.
Data Polarity			Not implemented at this time.
Clock Polarity			Not implemented at this time.
Tape Out	Selectable Setup State	NRZ-L NRZ-S NRZ-M BIO-L BIO-M BIO-S	Selects the output data format on the Tape Output pins of the connector.
DEMOD PARAMETERS			
Modulation		Disable BPSK QPSK	When disabled, no other Demod Parameters are applicable and all other demod fields appear on the screen as gray entries that can not be edited. No demodulation occurs when this parameter is disabled.
Nominal Freq			Expected input rate. Range is not limited.
Maximum Freq		On/off	This field specifies the maximum frequency that the demod will attempt to track; and is set up with both a check box and an entry field. The check box determines if the Maximum Frequency is based on the user-configured value entered in the entry field or if it is calculated automatically based on the LBW values. This value is not range limited.
Minimum Freq		On/off	This field specifies the minimum frequency that the demod will attempt to track; and is set up with both a check box and an entry field. The check box determines if the Minimum Frequency is based on the user-configured value entered in the entry field or if it is calculated automatically based on the LBW values. This value is not range limited.
Acquisition LBW		0 to 0.125	Specifies the LBW to be used while data is being acquired. This parameter is usually set wider than the Tracking LBW.

Parameter	Unit	Range or Values	Description
Tracking LBW		0 to 0.125	Specifies the LBW to be used after data is has been acquired and is continuous. This parameter can be set narrower than the Acquisition LBW to allow closer lock to the data. This value is also used to calculate the minimum and maximum frequency if the user elects to allow them to be calculated automatically.
BIT SYNC PARAMETERS			
Data Code		NRZ-L NRZ-S NRZ-M BIO-L BIO-M BIO-S	Selects the input data format to the bit synchronizer form the demodulator, if it is enabled.
Nominal Freq			Expected input rate. Range is not limited.
Maximum Freq			This field specifies the maximum frequency that the bit sync will attempt to track; and is set up with both a check box and an entry field. The check box determines if the Maximum Frequency is based on the user-configured value entered in the entry field or if it is calculated automatically based on the LBW values. This value is not range limited.
Minimum Freq			This field specifies the minimum frequency that the bit sync will attempt to track; and is set up with both a check box and an entry field. The check box determines if the Minimum Frequency is based on the user-configured value entered in the entry field or if it is calculated automatically based on the LBW values. This value is not range limited.
Acquisition LBW			Specifies the LBW to be used while data is being acquired. This parameter is usually set wider than the Tracking LBW. Range is not limited.
Tracking LBW			Specifies the LBW to be used after data is has been acquired and is continuous. This parameter can be set narrower than the Acquisition LBW to allow closer lock to the data. This value is also used to calculate the minimum and maximum frequency if the user elects to allow them to be calculated automatically. Range is not limited.
GRT CORRECTION			
Manual Control	Checkbox		Allows user defines values for pipeline delay (bit periods) and Offset delay (uSeconds).
Pipeline Delay (bits)	Bit periods		When the Manual Control checkbox is checked, the user can enter a specific pipeline delay. When Manual Control is unchecked, this field will display the current pipeline delay constant in use.
Offset (uSec)	Micro-seconds		When the Manual Control checkbox is checked, the user can enter a specific offset delay. When Manual Control is unchecked, this field will display the current offset delay constant in use.
BOARD-LEVEL PARAMETERS			

Parameter	Unit	Range or Values	Description
Device Selection	Selectable Setup State	Select Device, Device 1, Device 2, Device 3	Chooses which Demod/Bit Sync/Viterbi board is to be controlled. Note that the user could switch between boards using one instance of this module, but this is not recommended. When the desktop is saved, only the current settings, including which board is selected, is saved. This means that previous setups for other boards won't be retained for the next time that a desktop is loaded.
Times Board Programm-ed			Reports how many times the user has programmed the board. No input is necessary from the user.
Viterbi Button	Push button		Pressing the Viterbi button causes an extended options window to be displayed. This window is used to set Viterbi operating parameters.

Extended Options Window Definition

Pressing the Viterbi button causes an Extended Options window to open as shown below. The table that follows defines the Viterbi operating parameters.



Module 7-6: Extended AVTEC DEMOD BIT SYNC VITERBI MODULE Window

Module 7-7: AVTEC DEMOD BIT SYNC VITERBI MODULE Extended Window Parameters

Parameter	Unit	Range or Values	Description
VITERBI PARAMETERS			
Viterbi Enable	Selectable Setup State	Enable Disable	This parameter controls the Qualcomm Viterbi decoder chip. Setup is via a check box. Remove the check to disable. Insert a check to enable.
Viterbi Source	Selectable Setup State	Internal	Data can only be processed from the output of the bitsync.

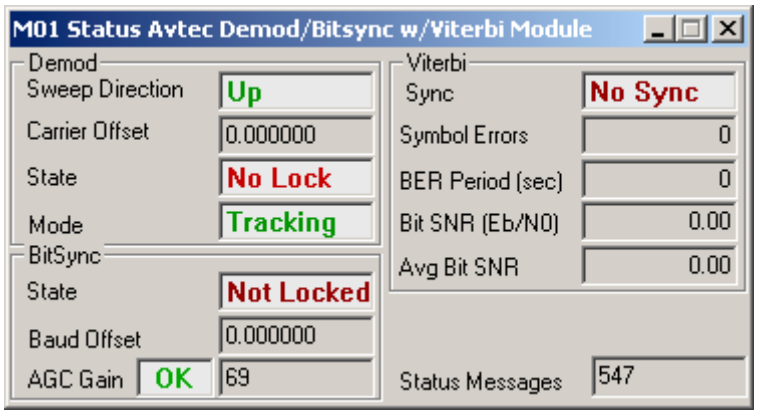
Parameter	Unit	Range or Values	Description
Invert Symbol A	Selectable Setup State	Enable Disable	Specifies data inversion on Symbol A. Setup is via a check box. Remove the check to disable. Insert a check to enable.
Invert Symbol B	Selectable Setup State	Enable Disable	Specifies data inversion on Symbol B. Setup is via a check box. Remove the check to disable. Insert a check to enable.
Swap A and B	Selectable Setup State	Enable Disable	Specifies if Symbol A and B are swapped. Setup is via a check box. Remove the check to disable. Insert a check to enable.
Auto BER Period for SNR Calc	Checkbox		Allows the BER calculation software to adjust the period from 1 to 10 seconds.
Decoder Control 1	Hexadecimal, 8 bits	00 to FF, default 04	Controls the value programmed for the Qualcomm Q-1900 Decoder Control 1 register.
Decoder Control 2	Hexadecimal, 8 bits	00 to FF, default 03	Controls the value programmed for the Qualcomm Q-1900 Decoder Control 2 register.
Decoder Control 3	Hexadecimal, 8 bits	00 to FF, default 01	Controls the value programmed for the Qualcomm Q-1900 Decoder Control 3 register.
T-Count	Decimal	256 to 65536 by 256, default 512	Controls the value programmed for the Normalization Test Bit Count Input Register of the Q-1900 Viterbi Decoder. Refer to the Q1900 Viterbi/Trellis Decoder documentation. The actual programmed value is displayed in hexadecimal next to the decimal input.
N-Count	Decimal	4 to 2044 by 8, default 52	Controls the value programmed for the Normalization Test Normalize Count Input Register of the Q-1900 Viterbi Decoder. Refer to the Q1900 Viterbi/Trellis Decoder documentation. The actual programmed value is displayed in hexadecimal next to the decimal input.
BER Period	Decimal seconds		Controls the length of the BER test period. Refer to the Q1900 Viterbi/Trellis Decoder documentation. The actual programmed value is displayed in hexadecimal next to the decimal input. This field is only active when the Auto BER Period checkbox is unchecked.
Min BER Period	Decimal seconds		Controls the minimum length of the BER test period. Refer to the Q1900 Viterbi/Trellis Decoder documentation. This field is only active when the Auto BER Period checkbox is checked.
Max BER Period	Decimal seconds		Controls the maximum length of the BER test period. Refer to the Q1900 Viterbi/Trellis Decoder documentation. This field is only active when the Auto BER Period checkbox is checked.
Min Error Threshold (bits)	Decimal bits	Default 100	Controls the minimum error threshold of the BER test period. Refer to the Q1900 Viterbi/Trellis Decoder documentation. This field is only active when the Auto BER Period checkbox is checked.

Configuration Notes

There are no configuration notes for the AVTEC DEMOD BIT SYNC VITERBI MODULE.

Status Window Definition

When you click the Status button on the AVTEC DEMOD BIT SYNC VITERBI MODULE window, a Status window appears, as shown below.



Module 7-8: Status AVTEC DEMOD BIT SYNC VITERBI MODULE Window

Module 7-9: AVTEC DEMOD BIT SYNC VITERBI MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
DEMOD			
Sweep Direction	Current state of Card	Up, Down	Specifies the current direction that the demod is currently sweeping.
Carrier Offset			Displays the data offset from the carrier
State	Current state of Card	No Lock, Lock	Current lock state of the demod.
Mode		Acquisition, Tracking	Specifies which tracking state is currently active.
VITERBI			
Sync		Sync OK, No Sync	Reports Viterbi lock state as reported by the Qualcomm chip.
Symbol Errors	Decimal count		Count of symbol errors as reported by the Qualcomm chip.
BER Period (sec)	Decimal seconds		Current BER period.
Bit SNR (Eb/N0)	Float		Current Eb / N0.
Av Bit SNR	float		Average Eb / N0.
BITSYNC			
State		Lock, Not Locked	Bit sync reported state.
Baud Offset			Baud offset from the ideal.

Parameter	Unit	Range or Values	Description
AGC Gain		OK/ value	Shows an integer value from 0 to 255 as to AGC Gain. The value 255 usually indicates no input.
BOARD-LEVEL			
Status Messages		Increments by one starting at 0	Count of status messages received by the GUI.

Module 8 AVTEC RATE ADJUST MODULE

MODULE NAME: AvtecRateAdjust.dll

Key Functions

- Accepts bursty data input and outputs a contiguous bit stream at a configurable rate
- Provides Serial Output Module capabilities plus rate adjust capabilities and multiple frames per buffer transfer to hardware (the later provides better performance at high rates and smaller frame lengths)
- Generates an idle pattern output when input is not available
- Provides configurable rate slewing, i.e., provides configurable depth and frequency of each rate adjustment

Required Hardware

- This module requires that one of the following boards be installed in the Avtec PTP for Windows system:
  - ✓ Monarch
  - ✓ Monarch E
  - ✓ AT-HSIO2
- The Reed-Solomon and convolutional encode functions require a Monarch E board

Add Module Guidelines

One module per board in the system can be setup. The 32 modules per desktop limit cannot be exceeded.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from any module: typically sources are the deblocker or data simulation source

CONNECTIONS: Output is to the board hardware, which is available on the output connector or for loopback.

EVENTS: Generates events at end-of-frame length, which is used when generating fixed rate data (simulator or BERT).

Module Functional Description

The AVTEC RATE ADJUST MODULE accepts bursty data input and outputs a contiguous bit stream that may include idle frames to fill input data gaps. In addition to generating fill frames to maintain a contiguous output, the onboard transmit frequency is dynamically adjusted to match the rate of the incoming data. To evaluate the input rate, the RATE ADJUST MODULE creates a queue of data buffers in system memory and monitors the amount of data

that is in the queue. Starting from a configured output rate, the transmit clock frequency is adjusted up or down if the queue has less or more than a "middle" amount of data (configuration parameter "que mid").

Most functions provided by the Serial Output Module are also provided by the RATE ADJUST MODULE. Therefore, other than queue and rate adjust parameters, output parameters are identical to the Avtec Serial Output Module. The additional setup information includes buffer depth control and rate damping control.

The RATE ADJUST MODULE is typically used with the NASCOM Deblocker or Scatter/Gather Modules to reconstruct a serial data stream that was encapsulated for transmission across a network.

Like the Serial Output Module, the RATE ADJUST MODULE communicates with the I/O board using the AV_FSTS Frame Synchronizer/Telemetry Simulator driver for Windows NT. The AV_FSTS driver refers to the boards by channel number *(Channel 0, 1, etc.).* The mapping between AV_FSTS channel number and physical I/O board is defined in the Windows NT registry using the FSTS_CONFIG.EXE utility.

Input Data Format

Input is either fixed or variable buffer lengths. If input is from the deblocker, then the input is variable length: it changes depending on the size of the data extracted from the blocks data field.

Output Data Format

Output reflects any rate adjusting that is required and may include idle frames. Idle frame length is defined by setup parameter.

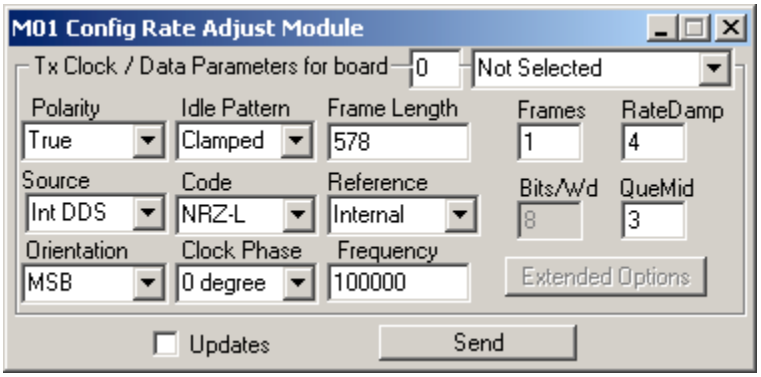
Module Window

When AvtecRateAdjust.dll is added to a desktop, the Avtec Rate Adjust dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 8-1: AVTEC RATE ADJUST MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AVTEC RATE ADJUST MODULE Board Parameters sub-menu.



Module 8-2: Config AVTEC RATE ADJUST MODULE Window

Module 8-3: AVTEC RATE ADJUST MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
TX CLOCK/DATA PARAMETERS-- Note that the module uses a software queue and adjusts output clock-- queue depth is decreasing than rate must be decreased-- speed clock up if queue is filling.			
For Board	Selectable Setup	Reflects boards in system/ FSTS_CONFIG.EXE	Defines which board is setup and controlled by this Rate Adjust Module.
Polarity	Selectable Setup	Inv True	Configures the output Data Polarity: Sets up module for True or Inverted output of the input data
Idle Pattern	Selectable Setup	Clamped Toggle	Specifies the idle that is transmitted for output idle frames. Idle frame size is determined by the Frame Length configuration. Idle frames are generated to maintain contiguous data output when there is a break in the incoming data. 1) Toggle: idle pattern is Alternating 1's and 0's 2) Clamped: the idle pattern has no transitions (true polarity is all zeros output-- inverted data means all ones output)
Frame Length	Words or bytes	Input frame length	Frame length in words (or bytes if bits per word = 8) The length should be configured close to input frame length (module can accept varying frame length, so this should be an average expected input)
Frames	frames		Number of frames per buffer transferred to the hardware for output For rate adjust, do not set for any other value than one The value entered affects performance. A larger number increases latency. Leave value at one for rates less than 1 Mbps-More frames per buffer means less calls to the driver and this should be used at higher rates. Values other than one are Intended for use with high rates or small frame lengths (high frame-rate interrupt)

Parameter	Unit	Range or Values	Description
Rate Damp	Definition of rate increment	0 to 32	This value defines the rate slewing, or the frequency (in occurrence) and size of each rate adjustment. The value determines how quickly the output clock adjusts based on input data rate and starting from the configured output data rate. Valid values are 0 – 32: 0 = fixed rate, 1 = minimum number of steps (big increments in rate adjust), 32 = maximum number of steps (small increments in rate adjust) 0 disables the rate adjuster: no rate adjusting-- it outputs whatever rate it is set for The bigger the number the slower the slewing.
Source	Selectable Setup	Internal DDS, Internal PLL, Ext TTL, Ext 422	Defines output clock source Do not use Internal PLL or external clock for rate adjust mode because they are fixed clock frequencies that cannot be adjusted. Must use internal DDS if module being used to rate adjust.
Code	Selectable Setup	NRZ-L, NRZ-M, NRZ-S, BI0-L, BI0-M, BI0-S	Output code type. Module converts the supplied data to the setup: if NRZ-L is chosen, then there is no conversion because this is the supplied format.
Reference	Selectable Setup	Internal, External TTL, External 422	Reference source for the Internal DDS: Internal, External TTL, or External RS-422 Output divisor picked from this reference frequency
Bits/Wd	N/A	N/A	Not user-configurable Number of bits per word (4-16), default = 8
QueMid	buffers		Minimum number of buffers that Rate Adjust Module will try to maintain in queue At start of data input, module does not transmit data until the queue reaches Que Mid (idle pattern is initially output), then data transmission begins and the module adjusts clock to try and keep queue at this mid point. If a queue underflow occurs, then the module stops data output, transmits idle, and does not begin data transmission again until the queue reaches Que Mid value.
Orientation		MSB, LSB	MSB first or LSB first defines how data is output-- defines if it does a bit reversal of the input data
Clock Phase		0 degree, 180 degree	Defines how clock is output 0 degrees (data transitions on rising edge of the clock) or 180 degrees (data transitions on the falling edge of the clock)
Frequency	Hz		Output Data Rate in Hz. Defines the start frequency from which module adjusts up and down

Extended Options Window Definition

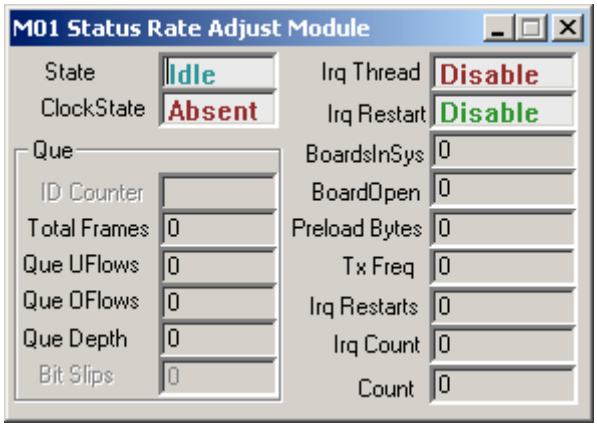
There is no extended options window for the AVTEC RATE ADJUST MODULE.

Configuration Notes

There are no configuration notes for the AVTEC RATE ADJUST MODULE.

Status Window Definition

When you click the Status button on the AVTEC RATE ADJUST MODULE window, a Status AVTEC RATE ADJUST window appears, as shown below.



Module 8-4: Status AVTEC RATE ADJUST MODULE Window

Module 8-5: AVTEC RATE ADJUST MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
GENERAL STATUS			
State		Idle, Transmit	Data output state
Clock State		Absent, Present	The state of the selected output clock When using external clock source, an absent here indicates that there is nothing applied to ST
Irq Thread		Disable, Enable	Displays Rate Adjust Output Thread state Indicates if Avtec PTP for Windows software is enabled or disabled
Irq Restart		Disable, Enable	Displays Automatic Restart State. Enabled on error.
Boards InSys		Reflects number of boards	Displays number of AT-HSIO2 and MONARCH-E boards in system as recorded in registry
BoardOpen		Reflects configuration	Specifies current board in use-- refer to configuration setup
Preload Bytes	bytes	N/A	N/A
Tx Freq	Hz	Reflects output	Current output frequency in Hz after rate adjust - the actual rate adjusted frequency

Parameter	Unit	Range or Values	Description
Irq Restarts	Restarts	Incrementing count	Count of Rate Adjust Serial Output thread restarts on error For rate adjust mode, this does not indicate corrupted data
Irq Count	Interrupts	Incrementing count	Count of interrupts
Count	frames	Incrementing count	Count of frames written to AV_FSTS driver
QUE			
ID Counter	N/A	N/A	(Future)
Total Frames	frames	Incrementing count	Count of frames transmitted Includes idle frames that are of defined frame length. Real data can be variable frame length-- For deblocker input application, this value equals number of blocks deblocked plus idle frames
Que UFlows	Under-flows	Incrementing count	Number of occurrences of no data If que Uflow occurs, idle pattern will occur in output data
Que OFlows	Over-flows	Incrementing count	Number of occurrences that the Rate Adjust went over the maximum allowable number of buffers A queue overflow indicates that data has been lost-- amount lost is unknown
Que Depth	frames		Number of buffers open Reflects the number of frames currently in the queue
Bit Slips	N/A	N/A	Not applicable

Module 9 AVTEC SERIAL INPUT MODULE

MODULE NAME: AvtecSerialInput.dll

Key Functions

- CCSDS and TDM Frame Synchronizer
- 64-bit Sync pattern and mask
- Programmable acquisition strategy
- Auto Polarity Detection and Correction
- Time Tag Frames
- Generate and Append Status
- Measures Bit Rate
- Asynchronous Bit Stream Throughput

(PTPNT-6100 Monarch-E features)

- CCSDS Pseudo-Derandomizer
- Reed-Solomon Decoder (header and frame)
- CRC Decoder

Required Hardware

- At least one of the following boards must be installed:
  - ✓ Avtec Monarch (PTPNT-6200 or 6205)
  - ✓ Avtec Monarch E (PTPNT-6100)
  - ✓ Avtec AT-HSIO2 (PTPNT-3037)

- Avtec Monarch E required for Reed-Solomon Decode function.

- Apogee Time board required for ERT time stamp using external time. ERT time stamp using PC time does not require Apogee Time board. The standard time tag accuracy is 100 microseconds. The Monarch and Monarch-E boards can provide 5 microsecond time tag accuracy with the –T option (PTPNT-6100-T or PTPNT-6200-T). With the –T option, the Monarch (E) strobe output must be connected to the Apogee ISA-STG2 mailbox input. The ISA-STG2 mailbox to Monarch (E) board mapping is defined using the FSTS_Config utility.

Add Module Guidelines

One Serial Input Module can be added to a desktop for each Monarch, Monarch E or AT-HSIO2 present in the system. Use the Serial Input Module configuration window ("Board Parameters for board" field) to identify which card is being set up by the module. Each card can simultaneously be set up with one Serial Input Module and one Serial Output Module.

Required Input Source, Connections, and Events

INPUT SOURCE: A serial bit stream input must be supplied to the card associated with the module (Monarch, Monarch E or AT-HSIO2). The input must be supplied either via the card's external DB-37 connector or using the loopback

function that is part of the Serial Input Module's functionality. If the loopback function is being used, then a Serial Output Module must also be set up for that same board, and it must be setup to provide the expected input.

CONNECTIONS: This module requires no input module connections since the input data is supplied externally. This module outputs data (frame per frame data) on port one. Unless port one is connected to the input of another module, the data is dropped. To provide internal test functions, two other ports provide data output. Port 3 outputs the entire buffer as it was received from FSTS driver (may be multiple frames per buffer) plus status. It keeps outputting as long as it receives data. Port 2 outputs the very first frame that it receives from the driver.

EVENTS: Although rarely used, this module generates an event at the end of every received frame: every time the driver returns a buffer of data, an event is generated. There is currently no standard use for this event in operational scenarios. The even is not generated unless it is connected to an event destination.

Module Functional Description

The AVTEC SERIAL INPUT MODULE supports the Frame Synchronizer portion of the AT-HSIO2 and Monarch and it supports the Frame Synchronizer/Reed-Solomon Decoder portion of the MONARCH-E. The Serial Input module reads frame data from the I/O board and outputs buffers with frame data and quality annotation to other modules.

Refer to the AT-HSIO2 or MONARCH-E Functional Descriptions for more detailed information on particular parameters.

The Serial Input Module communicates with the I/O board using the AV_FSTS Frame Synchronizer/Telemetry Simulator driver for Windows. The AV_FSTS driver refers to the boards by channel number *(Channel 0, 1, etc.).* The mapping between AV_FSTS channel number and physical I/O board is defined in the Windows registry using the FSTS_CONFIG.EXE utility.

Input Data Format

Input data must be a serial bit stream of data.

Output Data Format

Frame Output

When setup for asynchronous throughput, the output is a serial bit stream of data. When set up to frame synchronize, the output is frame data. Frame output can optionally include status appended to each frame.

Depending on the module setup, output frames may be CRC checked and/or Reed-Solomon decoded and corrected. The optionally appended status reflects the results of both functions.

Annotation Format

The Serial Input Module appends status information to every frame. The appended status is up to 48 bytes long and includes 12 bytes for Time Tag and Frame Synchronizer Status and 32-bytes for RS Decoder Quality Annotation (MONARCH-E). The appended status is available to other modules to create headers or trailers with specific data quality and time information. The appended status can also be recorded to disk along with the frame data using the File Recorder Module. When using the Serial Input Module Output Port 1, the appended status is passed to other modules in a buffer descriptor so it is not part of the actual data buffer. When using the Serial Input Module Output Port 3, the appended status is also passed in the data buffer as it is received from the FSTS driver. The tables below define the format of the appended status when using Output Port 3. These tables also provide a detailed definition of the status that is made available to other modules through the buffer descriptor when using Output Port 1.

The following five tables define the annotation format. The first table provides the top-level format overview that defines the four types of status information provided: Application Flags, Time Stamp, Hardware Flags, and RS Annotation. The Reed-Solomon status applies only to the MONARCH E board. The second table defines the PB4 time format.

The three remaining tables provide the byte and bit-level definitions for each of the four status areas. The second table defines the 32-bit Hardware Flags field in the Appended Status. The third table defines the 32-byte Reed-Solomon Decoder Quality Annotation. The fourth table supplements the previous table and provides the details of the Reed-Solomon Decoder Terse Quality field.

Module 9-1: MONARCH and MONARCH-E 48-byte Appended Status Format

Parameter	Size	Description
Application_Flags	4 bytes	If "Use Time Board" is selected, then this field contains a 32-bit count of the number of ticks accumulated while the frame was received (Monarch only). This field is in Big-Endian format (the most significant byte is the first byte in the field). This field is used by the Serial Input Module to compute the input bit rate. If "Use Time Board" is not selected or if the AT-HSIO2 board is in use then this field is undefined.
Time Stamp	8 bytes	If "Use Time Board" is selected, then the first six bytes of this field contain the frame time tag in PB4 format. If "Use Time Board" is not selected, then the last 4 bytes of this field contain the 32-bit tick counter from the Monarch or Monarch-E board.

Parameter	Size	Description
Hardware Flags	4 bytes	Bit mapped hardware status generated by the MONARCH Frame Synchronizer. Details are provided in the table below. This field is in Big Endian format (the most significant byte is the first byte in the field)
RS Annotation	32 bytes	Quality Annotation generated by the Reed-Solomon Error Correction chip. Details are provided in the table below. These status fields are in Big-Endian format. (All zeroes for MONARCH)

Module 9-2: NASA PB-4 Time Code Format

Item	Field Name	Format & Size	Value
1	Days of Year	Unsigned integer (11 bits)	Range 1 to 365
2	Milliseconds of Day	Unsigned integer (27 bits)	Range = 0 to 86399999
3	Microseconds of a Millisecond	Unsigned integer (10 bits)	Range = 0 to 999
	Total Length	6 bytes	

Module 9-3: 4-Bytes Hardware Flags Status Format

Bit(s)	Name	Description
31:30	SICQ<1:0>	This is a 2 bit field that indicates the state of the serial input channel control logic after the frame was received. Values are defined as follows: SICQ<1:0> Input Channel State 00 search (a dropout occurred) 10 flywheel 11 lock
29	APC_FLG	This is the Autopolarity Correct Flag. When it is high (1), it indicates that the serial input control logic locked on an inverted synchronization pattern and that the Autopolarity Correct logic is active.
28	CRC_ERR_FLG	This is the CRC Error Flag. It is set high (1), if a CRC error occurs on that frame. A CRC error indicates that the Serial Input calculated CRC did not match the CRC value in the frame field. (This flag is not currently implemented)
27	SLIP_FLG	This is the Slip Flag. When it is high (1), it indicates that the previous frame is either too long or too short meaning that the Sync Pattern for the following frame arrived too early or too late.
26:24	SDIS<2:0>	This is the 2 bit Slip Distance. This field indicates the magnitude and direction of the current frame's slip. To calculate the slip, use the allowable slip window (Serial Input Config Synchronizer Window) The slip value is given by SDIS <2:0> - allowable slip window . For example, if the allowable slip window is 1 and SDIS <2:0> is 1, then no slip occurred. If the allowable slip window is 1 and SDIS <2:0> = 0, then the sync pattern occurred one bit later than expected.
23	X	Undefined.

Bit(s)	Name	Description
22	UNLOCK_FLG	This is the Unlock Flag. If this bit is high (1), it indicates that the serial input channel logic has entered the Search state from a Lock or a Flywheel state. It indicates loss of frame synchronization. This flag is not currently implemented. The same condition is indicated by a 5/CQ <1:0> value of 00.
21	LOCK_FLG	This is the Lock Flag. If high (1), it indicates that the Serial Input Logic has entered the Lock state from either Search or Check state. Will only be set for the first frame received after entering the Lock state.
20:16	X	Undefined. Sit to all ones (0x1F).
15:0	X	Not used. Set to all zeroes (0x0000).

Module 9-4: 32-byte Reed-Solomon Decoder Chip Annotation Format

Parameter	Size	Description
User Annotation [1]	4 bytes	32-bit user defined annotation from Serial Input Config Extended Options
User Annotation [2]	4 bytes	32-bit user defined annotation from Serial Input Config Extended Options
Frame Counter	4 bytes	32-bit count which is incremented for each frame passed through the RS decoder chip. Starting value is zero on enable.
Routing Data	1 byte	Not Used.
Terse Quality	1 byte	Contains high level information about the current frame. Defined in the table shown below.
Errors Corrected	1 byte	Number of RS (255,223) errors corrected
Header Errors Corrected	1 byte	Number of RS (10,6) header errors corrected
Errored Codewords	1 byte	Number of codewords with errors in the current frame. An uncorrectable codeword is also errored, and this is counted in both fields.
Uncorrectable Codewords	1 bytes	Number of uncorrectable codewords in the current frame.
Error Bitfield	2 bytes	16-bit field that indicates which codewords were errored. Bit 0 refers to the first codeword, bit 1 refers to the second, etc. As with the previous quality annotation fields, if the uncorrectable bit is set for a certain codeword, the same bit is set in the errored bit field.
Uncorrectable Bitfield	2 bytes	16-bit field that indicates which codewords were uncorrectable. Bit 0 refers to the first codeword, bit 1 refers to the second, etc.
Bit Errors	2 bytes	Total number of bit errors found and corrected by the RS (255,223) decoder. The block decoder can correct up to 16 symbol errors per codeword. Because each symbol is a byte, the decoder could possibly correct 128 bits in a single codeword. This field reports the total number of bits corrected by the block decoder.
Errors Per Codeword	8 bytes	16-nibble fields; each nibble indicates the total number of errors per codeword. The first nibble refers to codeword 16, and the last nibble refers to codeword 1.

Module 9-5: Reed-Solomon Decoder Chip Terse Quality Definition (supplement to previous table)

Bit #	Description

Bit #	Description
7	Long frame
6	Short frame
5	Unrouteable frame
4	Uncorrectable frame
3	RS (255,223) uncorrectable frame
2	RS (10,6) uncorrectable header
1	RS (255,223) found errors in frame
0	RS (10, 6) found errors in header

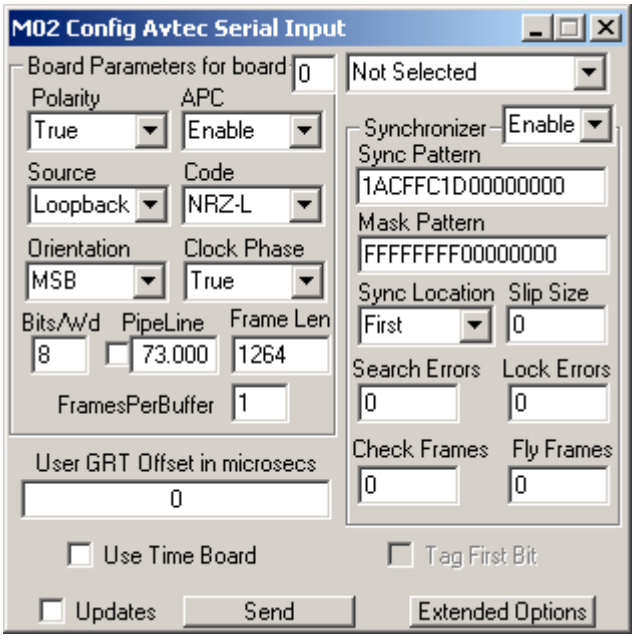
Module Window

When *AvtecSerialInput.dll* is added to a desktop, the AVTEC Serial Input dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 9-6: AVTEC SERIAL INPUT MODULE Window

Configure Window Definition

When you click the Configure button, a Config AVTEC SERIAL INPUT window appears, as shown below. The configuration parameters shown in the table below define the fields in the Config AVTEC SERIAL INPUT and apply to both the AT-HSIO2 and the MONARCH-E.



Module 9-7: Config AVTEC SERIAL INPUT Window

Module 9-8: Config AVTEC SERIAL INPUT Board Parameters

Parameter	Unit	Range or Values	Description
The entry after 'Board Parameters for board' defines which AV_FSTS channel, and thus which I/O board, is assigned to this module. In the figure above, this value is 0 which means that no board is assigned to the module. **The board number should be set to the desired AV_FSTS channel number plus one. For example, to assign the AV_FSTS channel 1 receiver to the serial input module, set the board number to 2. The list box to the right of the board number can also be used to assign a board to this module. The list box indicates whether the selected board is an AT-HSIO2, MONARCH, or MONARCH-E.**			
Polarity	Selectable Setup State	True Inverted	The set up defines the expected board input data as either true or inverted.
APC	Selectable Setup State	Enable Disable	If enabled, it automatically corrects the data polarity based on the detected sync pattern polarity. If an inverted sync pattern is detected and APC is enabled, the APC logic inverts the data. The Serial Input Status Polarity field will be Normal if the APC is not active and Inverted if the APC logic is active.
Source	Selectable Setup State	Ext TTL, Ext 422, Loopback	Defines the source of the input clock and data. For Ext TTL and Ext 422, the input clock and data are taken from the board's DB37 connector. For Loopback, the board output clock and data are internally connected to the input.
Code	Selectable Setup State	NRZ-L, NRZ-M, NRZ-S, Biphase-L, Biphase-M, Biphase-S	Defines the input PCM data type. The AT-HSIO2 and MONARCH-E support Biphase codes by delaying the clock with respect to the data and will only work for Loopback. A Bit Synchronizer is required to support external Biphase data sources.
Orientation	Selectable Setup State	MSB LSB	Defines the operation of the serial to parallel converter. Select MSB for data that was transmitted Most Significant Bit first. Select LSB for data that was transmitted Least Significant Bit first.

Parameter	Unit	Range or Values	Description
Clock Phase	Selectable Setup State	True Inverted	If True is selected, the Serial Input hardware expects the input data to change on the rising edge of the clock and it will latch the data on the falling edge of the clock. If Inverted is selected, the Serial Input hardware expects the input data to change on the falling edge of the clock and it will latch the data on the rising edge.
Bits/Wd	bits	4 to 16	Provides a default value of eight. Any value from 4 to 16 can be entered. The setup defines the word size that the hardware uses for serial to parallel conversion. For CCSDS telemetry streams, the default value of eight should be used. If Bits/Wd is less than eight, the input data will be stored in bytes. If Bits/Wd is greater than eight, the input data will be store in 16-bit words.
PipeLine	Float	Any positive floating point value	The Pipeline field can be used to correct the frame time tags for the pipeline delay in the Frame Synchronizer hardware. This field defines the number bit periods to subtract from the frame time tag. The bit period is calculated based on the Serial Input bit rate measurement. This value will be set automatically based on the frame synchronizer configuration. To override this automatic setting, click the check box next to the Pipeline entry box and enter a new PipeLine value. To disabled time tag correction, click the check box and set PipeLine to zero.
Frame Len	8-bit word OR 16-bit word	8 to 4096 OR 4 to 2048	This field defines the number of telemetry words in each frame input. If Bits/Wd is less than or equal to eight, then the Frame Len is equal to the number of bytes in the input frame. If Bits/Wd is greater than eight, then the number of bytes in the input frame is twice the Frame Len. The minimum frame length is 8 bytes. The maximum frame length is 4096 bytes.
Frames Per Buffer	Frames per buffer	Minimum value is 1: Maximum value is input data rate dependent	This field defines the number of telemetry frames that the Serial Input Module provides to other modules in a single buffer. The default value is 1 frame per buffer. If this value is greater than one, then the Serial Input Module will gather groups of consecutive telemetry frames before posting a buffer to other modules. This value should be set to keep the buffer rate under 1000 buffers per second. Frames Per Buffer = Bit Rate / (Frame Length in Bits * 1000).

Module 9-9: Config AVTEC SERIAL INPUT Synchronizer Parameters

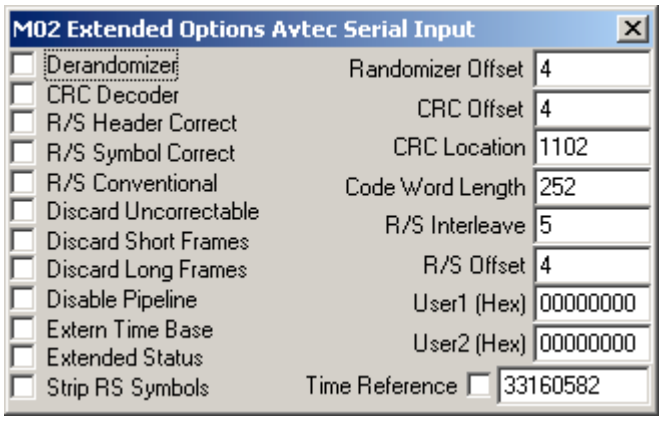
Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Synchronizer	Selectable Setup State	Enable Disable	Sets the card up to either synchronize to input data stream or to pass data through asynchronously. If Enable is selected, the Serial Input will only provide frame data buffers to Avtec PTP for Windows if an acceptable sync pattern is detected in the input data stream-- only synchronized data is output to the next module in the data flow, and all other data is dropped. If Disable is selected, all data is throughput. The Serial Input will perform serial to parallel conversion and provide data buffers to Avtec PTP for Windows as long as the input clock is active without any regard to frame boundaries.
Sync Pattern	Hex pattern	Any 64-bit pattern	Sixteen digit hexadecimal (64-bit) value that defines the frame synchronization pattern. All 16 digits should be entered even if they are zero. The actual bits used for synchronization are defined by the Mask Pattern. The default value of 1ACFFC1D00000000 is the 32-bit sync pattern for CCSDS telemetry streams.
Mask Pattern	Hex pattern	64-bit pattern	Sixteen digit hexadecimal (64-bit) value that defines the synchronization pattern mask. The mask determines which of the Sync Pattern bits must be matched on the input data. Each bit of the mask maps directly to the Sync Pattern starting from the left. A mask value of one indicates that the corresponding Sync Pattern bit must be matched. Sync Pattern bits whose corresponding Mask Pattern bits are zero are ignored by the input correlator. The correlator threshold is calculated as the number of ones in the Mask Pattern minus the number of allowable errors. The default value of FFFFFFFF00000000 is the 32-bit mask pattern for CCSDS telemetry streams.
Loc	Selectable Setup State	First Last	This field defines the location of the synchronization pattern within the frame. The synchronization pattern can either be the first bits of a frame (first) or the last bits of a frame (last). For sync pattern first, the Serial Input provides data buffers that begin with the sync pattern. The sync pattern and mask pattern should be left-justified in the 64-bit entry fields provided for configuration. For sync pattern last, the Serial Input provides data buffers with the sync pattern at the end of the buffer. The sync pattern and mask should be right-justified in the 64- bit entry fields provided for configuration. The configured Sync Pattern and Mask Pattern should represent how the pattern appears in the serial stream. The Serial Input does not bit reverse the sync pattern or the frame data.

Parameter	Unit	Range or Values	Description
Window	bits	-3 to +3	This field defines the size of the frame synchronizer bit slip window. The bit slip window defines the number of bit periods before and after the end of the frame in which the frame synchronizer will look at the output of the correlator. If the Window is set to 0, then the sync pattern for the next frame must occur immediately after the end of the current frame. The Bit Slip Window is programmable from 0 to +/- 3 bit periods.
Search Errors	bits	0 to 15	This field defines the number of allowable bit errors in the synchronization pattern during search AND check modes. If a frame synchronization pattern is within the acceptable bit error tolerance, it is recognized as frame: it is identified as either a search frame or a check frame depending on the mode. The number of Search Errors is programmable from 0 to 15.
Lock Errors	bits	0 to 15	This field defines the number of allowable bit errors in the synchronization pattern during lock AND flywheel modes. If a frame synchronization pattern is within the acceptable bit error tolerance, it is recognized as frame, and it is identified as a lock or a flywheel frame depending on the mode. The number of Lock Errors is programmable from 0 to 15.
Check Frames	frames	0 to 7	This is the number of consecutive frames that have a synchronization pattern within the Search Errors value and do not exceed the Window value that must occur in the data stream before the synchronization logic moves from check mode to lock mode. The value is frequently set to zero so that the synchronization logic moves immediately from search mode into lock mode. The number of Check Frames is programmable from 0 to 7.
Fly Frames	frames	0 to 7	This is the number of consecutive flywheel frames that can occur before the synchronization logic goes back to search. The synchronization logic goes into flywheel mode when it is in lock mode but receives a frame with synchronization pattern bit errors that exceed the Lock Errors value or a frame slip that exceeds the Windows value. The number of Fly Frames is programmable from 0 to 7.
Offset in Microseconds			This field allows the time tags to be offset. Any value entered is added to current time. To subtract time, enter a negative number.
Use Time Board			Check box setup: insert a check to enable. When enabled, time tags are generated using the Apogee Time Board. There is a special case when frames per buffer is divisible by 16. In this case, the first frame is time-tagged by the hardware, then the next fifteen time tags are calculated using a tic counter to calculate elapsed time from first frame.
Tag First Bit			Check box setup: insert a check to enable. When disabled, time tag is generated based on when last bit of frame is received. When Enabled, the time tag is generated based on when first bit of the frame is received

Extended Options Window Definition

When you click the Extended Options button in the Config AVTEC SERIAL INPUT window, the window shown below appears. The table that follows defines the fields in the AVTEC SERIAL INPUT Extended Options window. **The Serial Input Extended Options only apply to the MONARCH-E Frame Synchronizer with Reed-Solomon Decoder.**



Module 9-10: Config AVTEC SERIAL INPUT Extended Options Window

Module 9-11: Config AVTEC SERIAL INPUT Extended Options Parameters

Parameter	Unit	Range or Values	Description
Derandom-izer	Selectable Setup State	Enable, Disable	Controls CCSDS derandomization. Checked is enabled and the input data stream will be derandomized. Disabled (check removed) means no de-randomization is performed on the input data.
Randomizer Offset	bytes		Defines in bytes the randomizer offset from the start of the frame. The Randomizer offset is programmable from 0 to Frame Length - 1. For CCSDS telemetry streams the value of 4 should be used to exclude the 32-bit attached sync marker.
CRC Decoder	Selectable Setup State	Enable, Disable	Controls CCSDS CRC checking. Checked is enabled. Note that the MONARCH-E performs the CRC calculation and comparison prior to Reed-Solomon decoding. When CRC and Reed-Solomon decoding are used simultaneously, the CRC decoder may report CRC errors that are then corrected by the Reed-Solomon decoder. The CCSDS Virtual Channel Processor module can be used to perform CRC check in software after hardware Reed-Solomon decoding is performed by the MONARCH-E.
CRC Offset		0 to (CRC Location - 2)	Defines the CRC calculation start point offset. The two typical setups are 0 and 4. A 0 setup means CRC calculation begins as the start of the frame. A 4 setup means that a 32 bit synchronization pattern is excluded from the CRC calculation. The CRC Offset is programmable from 0 to CRC Location - 2.

Parameter	Unit	Range or Values	Description
CRC Location	bytes	0 to (Frame Length -2)	Defines in bytes the location of the 16 bit CRC remainder value. For CCSDS Grade-3 service (no Reed-Solomon check symbols), the 16-bit CRC is the last 2 bytes in the frame. For CCSDS Grade-2 service (Reed-Solomon check symbols are present), the 16-bit CRC is the last 2 bytes in the VCDU or the 2 bytes immediately preceding the Reed-Solomon check symbols. The CRC Location is programmable from 0 to Frame Length - 2.
R/S Header Correct	Selectable Setup State	Enable, Disable	Controls the correction of the VCDU header using the RS(10,6) decoder. Checked is enabled.
R/S Symbol Correct	Selectable Setup State	Enable, Disable	Is a tri-state check box. Off - no rs checking On - rs symbol correction-- detects, reports, and corrects errors Grayed-- rs check, report, but do not correct Controls the correction of the VCDU using the RS(255,223) decoder. Checked is enabled.
R/S Conven-tional	Selectable Setup State	Dual Conven-tional	Unchecked selects the dual-basis for the RS code as defined by the CCSDS. Checked selects the conventional basis.
Code Word Length	bytes	33 to 255	This field is not used. The Code Word Length is automatically computed from the Frame Length, R/S Interleave, and R/S Offset. CodeWordLength = (Frame Length – R/S Offset)/ RS Interleave. The MONARCH-E supports codeword lengths from 33 to 255.
R/S Interleave			Defines the RS code interleave depth. The MONARCH-E supports interleave depths from 1 to 8.
R/S Offset			Defines the start of the RS codeblock in the frame. To exclude a 32-bit sync pattern, this value should be set to 4.
User1 (Hex)			User-defined field that will appear in the appended quality annotation
User2 (Hex)			User-defined field that will appear in the appended quality annotation
Discard Uncorrect-able	Selectable Setup State	Discard, Pass-through	Controls the filtering of uncorrectable frames. Checked, the serial input will discard uncorrectable frames. Unchecked, the serial input will pass uncorrectable frames.
Discard Short Frames	Selectable Setup State	Discard, Pass-through	Controls the filtering of short frames. Checked, the serial input will discard short frames. Unchecked, the serial input will pass short frames.
Discard Long Frames	Selectable Setup State	Discard, Pass-through	Controls the filtering of long frames. Checked, the serial input will discard long frames. Unchecked, the serial input will pass long frames.
Disable Pipeline	Selectable Setup State	Enable Disable	Enables or disables the Reed-Solomon decoder pipeline. Checked is disabled. The Reed-Solomon decoder pipeline can be disabled for low data rate streams (less than 32 Kbps) to reduce the latency introduced by the decoder. This field should not be checked for processing high data rate streams.

Parameter	Unit	Range or Values	Description
Extern Time Base	Selectable Setup State	Aux TTL, Aux 422	This parameter determines the time base for the time stamp counter. When enabled, the external time source from either the AUX_TTL or AUX_422 input (on the DB37 connector) drives the onboard timer/counter. The 'Source' field in the Serial Input Config (Loopback, Ext TTL, or Ext 422) also controls the external time source. For the Loopback and Ext TTL settings, the AUX_TTL signal is selected for the external time base. For the Ext 422 setting, the AUX_422 input is selected for the external time base. When Extern Time Base is not checked, the PCI clock/4 drives the onboard timer/counter.
Time Reference	Hz	Enable Disable	Check box setup: insert a check to enable. Entry field applicable only when check box enabled. When enabled, it tells the module to use whatever value is typed in regardless of the PCI frequency saved in the Windows registry. When Disabled, the modules checks for a value in the Windows registry, if there's no registry value, it uses whatever value is typed in, but never uses a value of less than 1000. The registry value is entered using the FSTS_Config utility. This parameter defines the frequency (in Hz) of the clock that drives the onboard timer/counter. When 'Extern Time Base' is not checked, this is the frequency of the system PCI clock. When Extern Time Base is enabled, this value should be set to the frequency of the external time base which is typically 5 MHz or 10 MHz
Extended Status			This field is not used. The Serial Input module will always pass frame status to other modules in the buffer descriptor. The contents of the status will vary based on the selected board (AT-HSIO2, Monarch, or Monarch-E). To access the appended status directly, use Output Port 3.
Strip RS Symbols	Selectable Setup State	Enable Disable	When enabled, each frame is output with RS symbols stripped

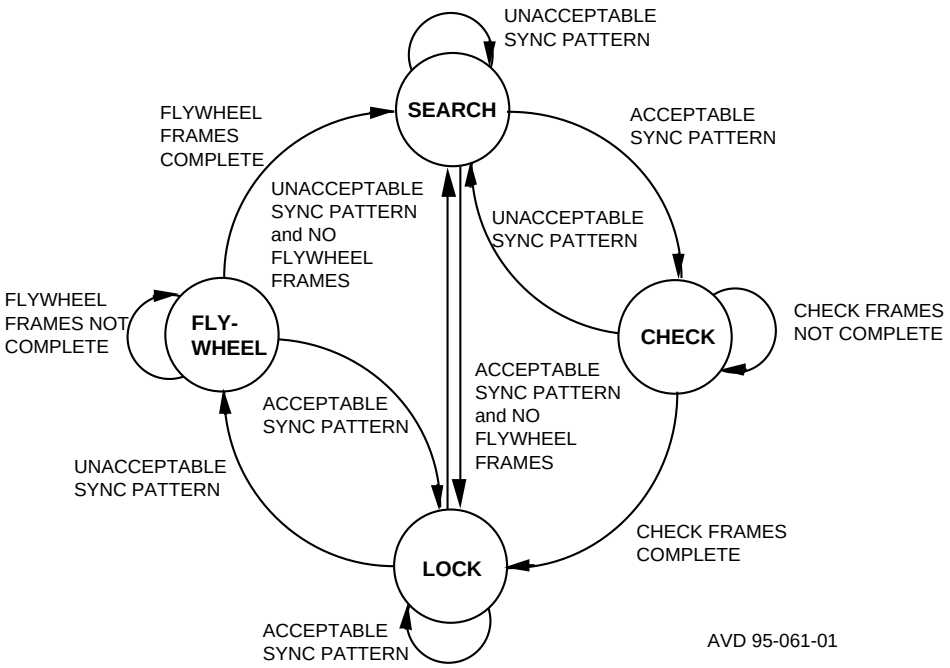
Configuration Notes

Frame Synchronization Logic Description

The Synchronizer sub-menu controls the operation of the I/O board frame synchronizer. The frame synchronizer control logic uses an adaptive strategy for acquiring minor frame synchronization which consists of four states: SEARCH, CHECK, LOCK, and FLYWHEEL.

A state diagram of the minor frame synchronizer control logic is shown below. While in the SEARCH state, the board searches the input data stream for an acceptable sync pattern. Once an acceptable pattern is found, the board enters the CHECK or LOCK state depending on the programmed number of check frames. The LOCK state is entered when N consecutive acceptable frames are received. The number of check frames N is programmable from 0 to 15 frames. If an unacceptable frame is received while in the CHECK state, the board returns to the SEARCH state. The board remains in the LOCK state as long as consecutive acceptable frames are received. If an unacceptable frame is received, the board advances to the

FLYWHEEL or SEARCH state depending on the programmed number of flywheel frames. If M consecutive unacceptable frames are received while in the FLYWHEEL state, then the board returns to the SEARCH state. The number of flywheel frames M is programmable from 0 to 15 frames. If an acceptable frame is received while in the FLYWHEEL state, then the board returns to the LOCK state. This method of operation ensures that the board will remain in LOCK even when the received data stream is corrupted by random bit errors.



Module 9-12: Frame Synchronizer State Diagram

Reed-Solomon Decoding

The MONARCH-E supports CCSDS Reed-Solomon codes with virtual fill and interleave depths up to 8. The following equation shows the relationship between frame length (Frame Len), Code Word Length, interleave depth (R/S Interleave), and unencoded offset (R/S Offset):

Frame Len = R/S Offset + (Code Word Length * R/S Interleave)

The amount of virtual fill in a frame is given by:

Virtual Fill = (255 - Code Word Length) * R/S Interleave

CRC Decoding

The MONARCH-E supports CRC decoding as defined in the CCSDS recommendations. The 16-bit CRC is the last two bytes in the VCDU. When Reed-Solomon coding is not used, the CRC location is Frame Length – 2. When Reed-Solomon coding is used, the CRC location is given by:

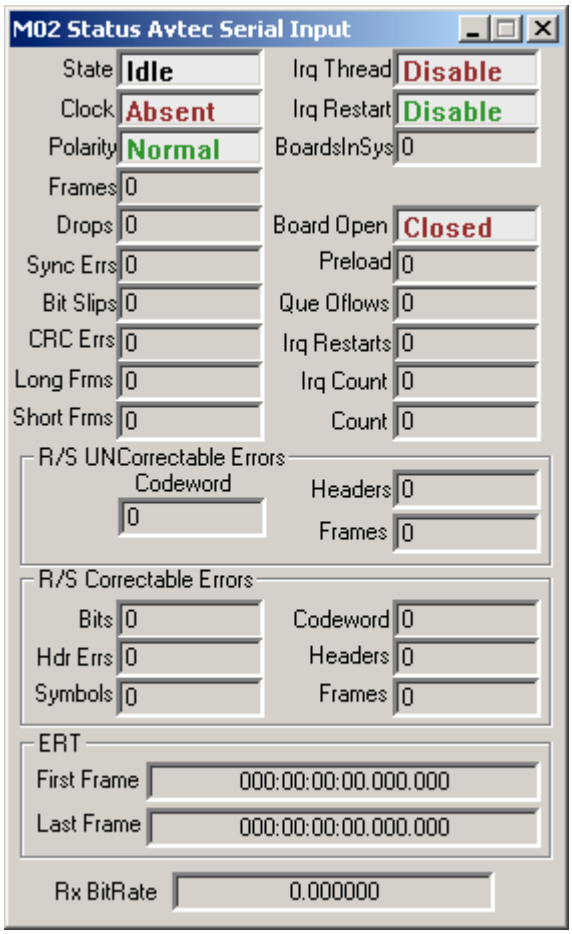
CRC Location = Frame Length – (32 * R/S Interleave) – 2.

A typical CCSDS transfer frame uses a Code Word Length of 252 bytes with R/S Interleave of 5. For this case, the equations above give:

Frame Len = 4 + (252 * 5) = 1264 bytes
CRC Location = 1264 – (32 * 5) – 2 = 1102.

Status Window Definition

When you click the Status button on the AVTEC SERIAL INPUT MODULE window, an AVTEC SERIAL INPUT window appears, as shown below. The tables below define the fields in the Status AVTEC SERIAL INPUT Window.



Module 9-13: Status AVTEC SERIAL INPUT MODULE

Module 9-14: Status AVTEC SERIAL INPUT Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
State	Sync logic summary	Idle Search Check Lock Flywheel	This parameter reflects the input synchronization logic and is detailed in the section above. A summary of each receiver state follows: 1) Idle- no data input 2) Search- data input and unable to synchronize to input pattern; searching for the sync pattern 3) Check- synchronization logic is in check mode. Search frame has been found and logic has progressed to check mode. Since the check mode is often zero frames long, the logic may skip directly to lock mode 4) Lock- synchronization logic is in lock mode. Input data matches setup synchronization pattern and frame length 5) Flywheel- synchronization logic is in Flywheel mode. Logic was in lock mode, but input data stopped matching sync pattern and frame length. Flywheel mode is frequently set to zero frames, which means logic moves immediately back to search mode.
Clock	Inputs clock status	Absent Present	Indicates whether the boards clock signal input is active (toggling).
Polarity		Normal Inverted	Indicates whether the Auto-Polarity Correction logic is active. The Polarity is Normal if the Auto-Polarity Correction logic is not active. The Polarity is Inverted if the Auto-Polarity Correction logic is active.
Frames	frames	increments	Count of frames received in the Lock or Flywheel States. Frames received in the Check State are not included in this count. To be recognized, a frame must meet all of the error boundaries defined by the Serial Input configuration. This means that a frame must not exceed the maximum allowable synchronization pattern bit errors and maximum allowable number of bit slips for the current synchronization mode in order to be counted as an input frame.
Drops	dropouts	increments	Count of dropouts (Lock to Search transitions).
Sync Errs			This status reports the total number of bit errors in the synchronization patterns of the frames recognized by the serial input. It is important to note that only frames with synchronization pattern bit errors within the tolerance are reported: frames that exceed the error tolerance were never recognized as frames and cannot be counted. (This field is not currently implemented).
Bit Slips			This status reports how many of the frames recognized were slipped either long or short. It is important to note that only frames with slip bit errors within the tolerance are reported: frames that exceed the error tolerance were never recognized as frames and cannot be counted.
CRC Errs			Count of frames received in the Lock or Flywheel state that has CRC errors. (MONARCH or MONARCH-E only)
Long Frms			Count of long frames as seen by the Reed-Solomon Decoder. (MONARCH-E only)

Parameter	Unit	Range or Values	Description
Short Frms			Count of short frames as seen by the Reed-Solomon Decoder. (MONARCH-E only)
Irq Thread			Indicates if this Serial Input Module is currently Enabled or Disabled.
Irq Restart			Indicates if the Serial Input Module is automatically restarting (Enable) because of an error. Should be disabled if no errors have occurred.
BoardsInSys			Count of AT-HSIO2 or MONARCH boards in system as recorded in the Windows registry
Board Open	state	Opened Closed	Displays if the board is currently Opened (Enabled) or Closed (Disabled).
Preload	bytes		Count of bytes that are preloaded into the AV_FSTS driver by the Serial Input Module.
Que Oflows	count		Count of overflows within the Serial Input thread. Should be zero for normal operation.
Irq Restarts	count		Count of Serial Input thread Restarts after overrun errors. A nonzero value indicates an error condition.
Irq Count	count		Count of successful attempts to read a buffer of data from the AV_FSTS driver.
Count	frame	Increments by one for each frame	Count of frames read from the AV_FSTS driver.
Rx Bit Rate	bps		This is the Received bit rate calculated by the module. Calculation is based on frame length and time stamp. Accuracy reflects the accuracy of the time source: Extern Time base or PCI clock.

The status values in the R/S UNCorrectable Errors sub-menu are reported by the MONARCH-E Reed-Solomon Decoder Chip.

Module 9-15: Status AVTEC SERIAL INPUT R/S Uncorrectable Errors Parameters

Parameter	Unit	Range or Values	Description
Codeword	codewords	Incrementing counter	Count of uncorrectable codewords (>16 byte errors)
Headers	headers	Incrementing counter	Count of uncorrectable headers (> 2 nibble errors)
Frames	frames	Incrementing counter	Count of frames with uncorrectable codewords

The status values in the R/S Correctable Errors sub-menu are reported by the MONARCH-E Reed-Solomon Decoder Chip.

Module 9-16: Status AVTEC SERIAL INPUT R/S Correctable Errors Parameters

Parameter	Unit	Range or Values	Description
Bits			Count of corrected bit errors
Hdr Errs			Count of corrected header errors - RS(10,6) code
Symbols			Count of corrected symbol errors
Codeword			Count of corrected code words
Headers			Count of corrected headers
Frames			Count of corrected frames

Module 9-17: Status AVTEC SERIAL INPUT Earth Receive Time (ERT)

Parameter	Description
First Frame	Time first frame received after Lock (Day of year: Hours: Minutes: Seconds : Milliseconds : Microseconds)
Last Frame	Time most recent frame received (Day of year: Hours: Minutes: Seconds : Milliseconds : Microseconds)

Module 10 AVTEC SERIAL OUTPUT MODULE

MODULE NAME: AvtecSerialOutput.dll

Key Functions

- Controls and monitors data output on one of the Avtec Telemetry Boards
- Configurable output mode: burst or contiguous bit stream

Required Hardware

- The AVTEC SERIAL OUTPUT MODULE requires one of the following Avtec boards be installed in the Avtec PTP for Windows system.
  - ✓ AT-HSIO2
  - ✓ Monarch
  - ✓ Monarch E
- A Monarch E Card is required for Reed-Solomon and convolutional encoding.

Add Module Guidelines

One AVTEC SERIAL OUTPUT MODULE per board can be included in a desktop. For each board, one Avtec Serial Input Module can be included in a desktop and setup to run simultaneously on the same card.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can supply the input data.

CONNECTIONS: Output from this module is via the DB-37 external connector.

EVENTS: Generates an event at end of each buffer (frame) output.

Module Functional Description

The AVTEC SERIAL OUTPUT MODULE supports the Telemetry Simulator portion of the AT-HSIO2 and the Reed-Solomon Encoder/ Telemetry Simulator portion of the MONARCH-E. The Serial Output module accepts buffers of data from other modules and writes the data to the I/O board for serial output. The Serial Output module can post an event each time it completes writing a buffer to the board.

Refer to the AT-HSIO2 or MONARCH-E Functional Descriptions (References #1 and 4) for detailed information on these boards.

The AVTEC SERIAL OUTPUT MODULE communicates with the I/O board using the AV_FSTS Frame Synchronizer/Telemetry Simulator driver for Windows NT. The AV_FSTS driver refers to the boards by channel number (Channel 0, 1, etc.). The mapping between AV_FSTS

channel number and physical I/O board is defined in the Windows NT registry using the FSTS_CONFIG.EXE utility.

Input Data Format

Any data type can be input.

Output Data Format

Bit contiguous (telemetry) or burst data output depending on module setup. Telemetry output can include idle frames. Telemetry output frame length is defined by Serial Output Module setup. Burst output frame length reflects the data source frame length.

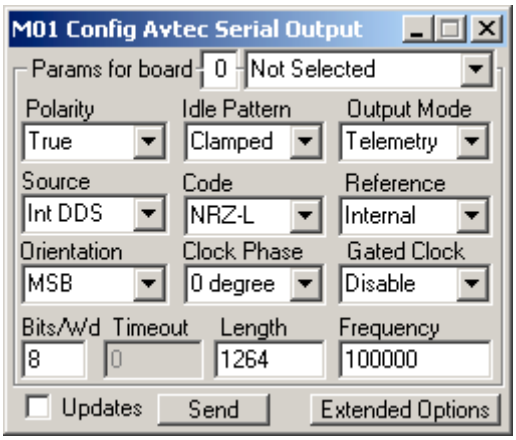
Module Window

When AvtecSerialOutput.dll is added to a desktop, the Avtec Serial Output dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 10-1: AVTEC SERIAL OUTPUT MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AVTEC SERIAL OUTPUT MODULE Board Parameters sub-menu.



Module 10-2: Config AVTEC SERIAL OUTPUT MODULE Window

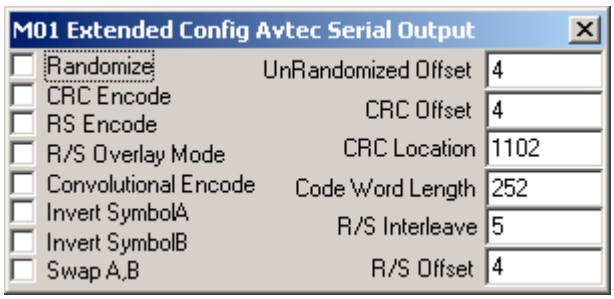
Module 10-3: AVTEC SERIAL OUTPUT MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
The entry after 'Tx Clock/Data Board Parameters for board' defines which AV_FSTS channel, and thus which I/O board, is assigned to this module. In the example Config window, this value is 0, which means that no board is assigned to the module. **The board number should be set to the desired AV_FSTS channel number plus one. For example, to assign the AV_FSTS channel 0 transmitter to the serial output module, set the board number to 1.**			
BOARD-LEVEL SETUP			
Params for Board	Selectable Setup State	Boards available in system	Defines board number and board type. This information identifies the Avtec board that is being controlled by the module.
Polarity	Selectable Setup State	True Inverted	Defines the output Data Polarity. If enabled, then data is inverted at output from the hardware.
Idle Pattern	Selectable Setup State	Clamped Toggle	Specifies idle transmission, all zeros (Clamped) or alternating 0/1 pattern (Toggle)
Output Mode	Selectable Setup State	Command, Safe Comm, Telemetry, Safe Telem	Defines the module output mode: 1) Command: Burst mode data output. Variable length on the output and the data source determines the output data frame length. 2) Safe Command: burst mode output, but module does not accept input when its buffers are full 3) Telemetry: Bit contiguous data output. If no data is available for output, it sends an empty frame that is defined by the idle fill pattern. Frame length is determined by the Serial Output configuration and the data supplied must match this setup. 4) Safe Telemetry: contiguous data output, but module does not accept input when its buffers are full Safe modes can be used for data flows where source is expecting acknowledgement when data received by Serial Output Module. Source won't receive an acknowledgement when buffers are full so that it knows to wait and then retransmit.
Source	Selectable Setup State	Int DDS Int PLL Ext 422 Ext TLL	Defines the clock source: Internal DDS or PLL External TTL or RS-422
Code	Selectable Setup State	NRZ-L NRZ-M NRZ-S BIO-L BIO-S BIO-M	Defines output. Module converts the supplied data to the setup: if NRZ-L is chosen, then there is no conversion because this is the supplied format. NRZ-L, M, S, and BI0-L, M, S
Reference	Selectable Setup State	Internal External TTL External 422	Identifies the reference for the DDS: Internal, External TTL or External RS-422

Parameter	Unit	Range or Values	Description
Orientation	Selectable Setup State	MSB LSB	MSB first, or LSB first
Clock Phase	Selectable Setup State	0 Degree 180 Degree	Defines clock output. 0 degree, data transitions on the rising edge of the clock; or 180 degree, data transitions on the falling edge of the clock
Gated Clock	Selectable Setup State	Disable Enable	Enable (Shuts off clock at underrun) or Disable (Continuous clock)
Bits/Wd	Bits/word		Number of bits per word, 4 through 16, default 8
Timeout	msec		msecs of no data until timeout (0 = no timeout)
Length	words		Frame Length in words (or bytes if bits per word = 8)
Frequency	Hz		Output data Rate in Hz

Extended Options Window Definition

The Serial Output Extended Options only apply to the MONARCH-E PCM Simulator with Reed-Solomon Encoder**.**



Module 10-4: AVTEC SERIAL OUTPUT MODULE Extended Options Window

Module 10-5: AVTEC SERIAL OUTPUT MODULE Extended Options Window Parameters

Parameter	Unit	Range or Values	Description
Randomize	Selectable Setup State	Enable Disable	When enabled, module randomizes data before output. All data except that defined by UnRandomized Offset is randomized. Checkbox setup: insert check to enable.
CRC Encode	Selectable Setup State	Enable Disable	Controls the CRC encoder. Checkbox setup: insert check to enable.
RS Encode	Selectable Setup State	Enable Disable	Controls the RS encoder. Checkbox setup: insert check to enable.

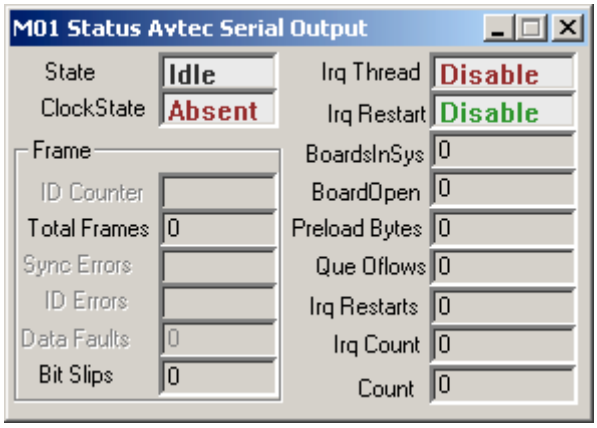
Parameter	Unit	Range or Values	Description
R/S Overlay Mode	Selectable Setup State	Enable Disable	The MONARCH-E RS encoder can append the RS check symbols (32 per codeword) to the output frame (not checked) or overwrite part of the output frame with the RS check symbols (checked) Overlay enabled means it receives a buffer (from data source module) that is large enough to include the RS symbols and the symbols are inserted Or if disabled, then the RS symbols are appended to the input buffer (the frame)
Convol-utional Encode	Selectable Setup State	Enable Disable	Controls the MONARCH-E rate 1/2, constraint length 7 convolutional encoder. Checked enables the convolutional encoder. Note the transmit clock rate corresponds to the symbol rate (or 1/2 the bit rate) when the convolutional encoder is enabled.
Invert SymbolA	Selectable Setup State	Enable Disable	Controls the inversion of the convolutional encoder SymbolA (or G1 = 1111001) output. Checked inverts G1 before output.
Invert SymbolB	Selectable Setup State	Enable Disable	Controls the inversion of the convolutional encoder SymbolB (or G2 = 1011011) output. Checked inverts G2 before output.
Swap A,B	Selectable Setup State	Enable Disable	Controls the order the convolutional encoder symbols are transmitted. Unchecked, SymbolA (G1) is transmitted before SymbolB (G2). Checked, SymbolB (G2) is transmitted before SymbolA (G1).
UnRandom-ized Offset	bytes	0 to frame length	Defines the randomizer offset from the start of the frame. To exclude a 32-bit sync pattern from randomization, this value should be set to 4.
CRC Offset	bytes	0 to frame length	Defines the starting point for CRC calculations. To include the sync pattern, set this value to 0. To exclude a 32-bit sync pattern, set this value to 4.
CRC Location	bytes	0 to frame length	Defines the location where the 16-bit CRC remainder should be written in the frame
Code Word Length	bytes		Defines the Reed-Solomon code word length from 33 to 255.
R/S Interleave	interleave		Defines the RS interleave depth from 1 to 8.
R/S Offset	bytes	0 to frame length	Defines the starting point of the RS codeblock within the frame.

Configuration Notes

There are no configuration notes for the AVTEC SERIAL OUTPUT MODULE.

Status Window Definition

When you click the Status button on the AVTEC SERIAL OUTPUT MODULE window, a Status AVTEC SERIAL OUTPUT window appears, as shown below.



Module 10-6: Status AVTEC SERIAL OUTPUT MODULE Window

Module 10-7: AVTEC SERIAL OUTPUT MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
MODULE SUMMARY STATUS			
State		Idle Transmit	Displays modules state: Idle-- no data output Transmit-- outputting data
ClockState			(future)
Irq Thread			Displays Serial Output Thread State
Irq Restart			Displays Automatic Restart State
BoardsIn Sys			Displays number of AT-HSIO2 or MONARCH-E boards in system as recorded in registry
BoardOpen			Specifies board in use
Preload Bytes			Number of bytes that are preloaded in the AV_FSTS driver
Que Oflows			Number of Queue Overflows in the Serial Output thread
Irq Restarts			Count of Serial Output thread restarts
Irq Count			Count of interrupts
Count			Count of frames written to the AV_FSTS driver
FRAME STATUS			
ID Counter			(not implemented)
Total Frames			Count of frames transmitted
Sync Errors			(not implemented)
ID Errors			(not implemented)
Data Faults			(not implemented)
Bit Slips			Number of bit slips (not implemented)

Module 11 AXAF SFDU FORMATTER MODULE

MODULE NAME: AxafSfduFormatter.dll

Key Functions

- Encapsulates data unit with 102 byte SFDU Header
- Adds pad at end of odd-byte length telemetry data field (applies to odd-byte frame input)
- Provides user-configurable header fields
- Automatically updates data-dependent header fields

Required Hardware

- The AXAF SFDU FORMATTER MODULE does not require any special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of AXAF SFDU FORMATTER MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is typically from either the Serial Input Module with extended status enabled or from the VC Processor module. Input required from VC Processor to satisfy all required status.

CONNECTIONS: This module outputs to port 1. Unless the output is connected to the input of another module, the data is dropped.

EVENTS: This module neither generates an event nor expects an event input.

Module Functional Description

The Advanced X-Ray Astrophysics Facility (AXAF) Standard Formatted Data Unit (SFDU) FORMATTER MODULE is a throughput encapsulation module that receives frame input and outputs unaltered frames in SFDU format.

The AXAF SFDU header format is based on the definitions as needed for support of the AXAF spacecraft. Frame synchronization states are recorded within the SFDU header only if a serial input module using the Monarch-E board is loaded as a source of the telemetry stream. In addition the Reed Solomon decoder and Extended Status must be enabled.

Note: See DSN Document 820-13, Rev. A – TLM-3-26 *DSN Telemetry Interface with MSFC for the Advanced X-ray Astrophysics Facility – Imaging (AXAF-I) Project* for a detailed overview of the AXAF SFDU format.

Input Data Format

The module expects CCSDS Version 1 or Version 2 frame input with Avtec PTP for Windows appended time stamp and status. Status from VC Processor is required to satisfy all required information.

Output Data Format

Outputs the input frame in SFDU Format. Format is a 104-byte header followed by telemetry data. The telemetry data is the input frame. Data unit must end on even byte boundary, so pad is added to odd-byte frames at the end of telemetry data.

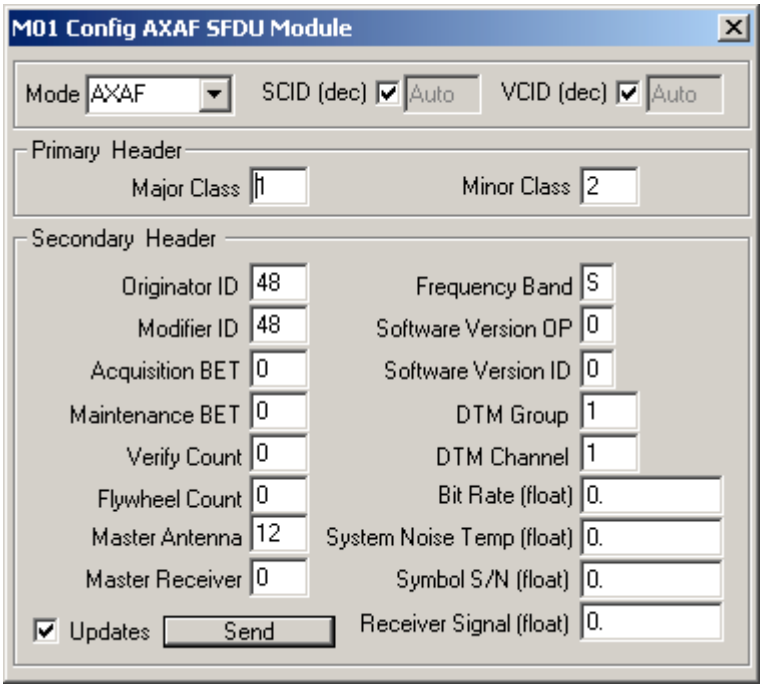
Module Window

When *AxafSfduFormatter.dll* is added to a desktop, the AXAF SFDU Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 11-1: AXAF SFDU FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AXAF SFDU FORMATTER MODULE Board Parameters sub-menu.



Module 11-2: Config AXAF SFDU FORMATTER MODULE Window

Module 11-3: AXAF SFDU FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
The AXAF SFDU header format is a 104 byte header that includes fixed fields, fields that reflect data input, and fields that are user-configurable. The setup fields provided in this window are only those fields that are defined by the operator. The range provided for each field is the number of bits assigned by the format definition. The description provides the header field name and some of the standard setup values used for that field. The operator can enter any value into the field that does not exceed the bit range. Default values show up when the window is brought up: for most fields the default value is zero.			
MODE CONTROL FIELDS			
Mode	Text	AXAF Standard	This selection controls extraction of the VCID from the frame header: it defines which and how many bits in the input frame header contain the VCID value. 1) Standard: The module reads the VCID value from the CCSDS-defined six-bit field in the input frame header. 2) AXAF: The module reads the VCID value from the top three bits of the CCSDS-defined six-bit field in the input frame header.
SCID	Decimal	Enable, Disable Text Entry	This field determines if the SCID value is read from the input frame or if the SCID value is read from the GUI entry: it defines which SCID value will be stamped into the AXAF header. When checked, the module is in automatic mode, which means that the SCID is read from the input frame. When unchecked (issue a send after removing the check so that the entry field is not grayed), the module will use the SCID value entered by the operator in the text field following the check box.

Parameter	Unit	Range or Values	Description
VCID	Decimal	Enable, Disable Text Entry	This field determines if the VCID value is read from the input frame or if the VCID value is read from the GUI entry: it defines which VCID value will be stamped into the AXAF header. When checked, the module is in automatic mode, which means that the VCID is read from the input frame. When unchecked (issue a send after removing the check so that the entry field is not grayed), the module will use the VCID value entered by the operator in the text field following the check box. **Sequence Count Note**: The sequence count (see status) reports the number of data units output from the module for that VCID. If VCID is a static value (every header uses the operator VCID entry), it affects the sequence counter because every frame output is from the same channel. Otherwise, the sequence count reflects the sequence count for *that* input VCID (this is the transmitted sequence count, not the frame header field).
PRIMARY HEADER			
Major Class	decimal	8-bit field Byte 29	Major Data Class; Default = 1 (Spacecraft telemetry data)
Minor Class	decimal	8-bit field Byte 30	Minor Data Class. Indicates configuration of DSCC (Deep Space Communications Complex) frame sync and Reed-Solomon decoding processes. Valid values = 0, 1, 2; AXAF-I, value=2
SECONDARY HEADER			
Originator ID	decimal	8-bit field Byte 37	Indicates message origin; 48=originated from DSN (Deep Space Network)
Modifier ID	decimal	8-bit field Byte 38	Indicates where message was last modified; 48=last modified by DSN
Acquisition BET	decimal	8-bit field Byte 55	Acquisition Bit Error Tolerance – defines number of allowable bit errors in sync pattern in search and verify modes; values=0 – 15
Maint-enance BET	decimal	8-bit field Byte 56	Defines number of allowable bit errors in sync pattern in lock and flywheel modes; values=0 – 15
Verify Count	decimal	8-bit field Byte 57	Number of frames required for frame sync logic to transition from verify to lock state; values=0 – 15
Flywheel Count	decimal	8-bit field Byte 58	Number of frames required for frame sync logic to transition from flywheel to search state; values=0 – 15
Master Antenna	decimal	8-bit field Byte 87	Indicates which DSS antenna data was received from.
Master Receiver	decimal	8-bit field Byte 88	Indicates which receiver was used with Master Antenna
Frequency Band	ascii	8-bit field Byte 66	Indicates frequency band of data; values = S, X, K

Parameter	Unit	Range or Values	Description
Software Version OP	ascii	8-bit field Byte 93	Software Version Operational Level; assigned by DSN Operations
Software Version ID	ascii	8-bit field Byte 94	Software Version Identifier; assigned by DTM CDE
DTM Group	decimal	8-bit field Byte 89	DSCC Telemetry Subsystem Group number
DTM Channel	decimal	8-bit field Byte 90	DTM Channel number
Bit Rate (float)	Float bps	32-bit field Bytes 67-70 Refer to float format in DSN Document 820-13	Measured bit rate of received telemetry data in bits per second
System Noise Temp (float)	Float kelvin	32-bit field Bytes 73-76 Refer to float format in DSN Document 820-13	System Noise Temperature in Kelvin
Symbol S/N (float)	Float dB	32-bit field Bytes 77-80 Refer to float format in DSN Document 820-13	Estimated signal-to-noise ratio (dB) in symbol domain (prior to Viterbi decoding)
Receiver Signal (float)	Float dBm	32-bit field Bytes 81-84 Refer to float format in DSN Document 820-13	Receiver signal level (dBm) for master receiver in array

Extended Options Window Definition

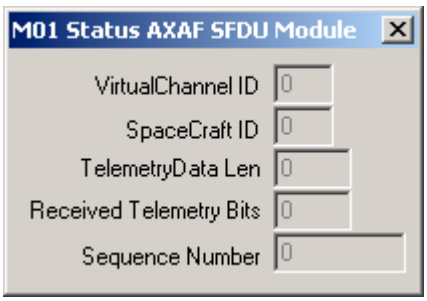
There is no extended options window for the AXAF SFDU FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the AXAF SFDU FORMATTER MODULE.

Status Window Definition

When you click the Status button on the AXAF SFDU FORMATTER MODULE window, a Status AXAF SFDU FORMATTER window appears, as shown below.



Module 11-4: Status AXAF SFDU FORMATTER MODULE Window

Module 11-5: AXAF SFDU FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Virtual Channel ID	VCID	Extracted from frame	Displays the VCID value of the input frame
SpaceCraft ID	SCID	Extracted from frame	Displays the SCID value of the input frame
Received Telemetry Bits	bits	Detected value from frame input	Input data length in bits (buffer length) Reports the length of the telemetry data portion that follows the SFDU header
Telemetry Data Len	bytes	Detected value from frame input	Equals input data length (buffer length) Reports the length of the telemetry data portion that follows the SFDU header. This value will reflect the frame length input to the module.
Sequence Number	Integer SFDU	Incrementing counter	Count of the number of SFDUs output for that virtual channel. Value is also put in SFDU header. Sequence number starts with one and increments for each SFDU output. Values are 1 to 2^32-1: after which the counter rolls over. **Note**: This is the transmitted count for all SFDUs of that channel. See configuration definitions to see how setup mode affects this count.

Module 12 AYDIN BIT SYNC MODULE

MODULE NAME: AydinBitSync.dll

Key Functions

- Controls Aydin Bit Sync card
- Receives and displays status information from the Aydin Bit Sync card.

Required Hardware

- Aydin Bit Sync card, model PC-335
- Appropriate cables to interface with the card.

Add Module Guidelines

Typically, one module is loaded per bit sync card in use. The quantity of bit sync cards supported is limited only by the number of available ISA bus slots in the computer. The user must have configured each instance of the card for the AUXIO driver. This is accomplished using the supplied AUXIO_CONFIG utility program, which generates the appropriate system registry information to control the Aydin cards. The cards are numbered sequentially, beginning with 1, as defined in the registry. The order of the cards electrically on the ISA bus is of no consequence.

Required Input Source, Connections, and Events

INPUT SOURCE: All data comes from external cables. No module should pass data to this module.

CONNECTIONS: No modules should be connected to this module.

EVENTS: No events are generated or required.

Module Functional Description

This module interfaces with the Aydin PC-335 card over the ISA bus, using the AUXIO driver. The user must have configured each card in the system registry using the AUXIO_CONFIG utility provided with the Avtec PTP for Windows software. Only control and status information is placed on the ISA bus; all user data in and data with clock is handled directly by the PC-335V card via external cabling. The card may be controlled by the console GUI or by Remote Interface Library (RIL) calls.

Input Data Format

The PC-335 accepts data in any of the NRZ (L,M,S) or BIØ (L,M,S) formats, see table below for a complete list.

Output Data Format

The user may select the desired output format. See table below.

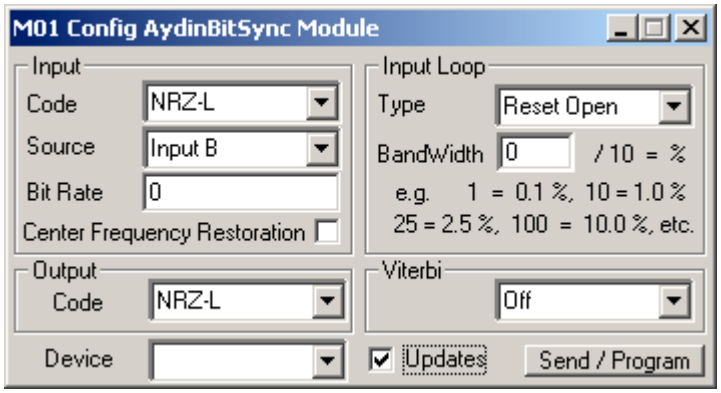
Module Window

When *AydinBitSync.dll* is added to a desktop, the Aydin Bit Sync dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 12-1: AYDIN BIT SYNC MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AYDIN BIT SYNC MODULE Board Parameters sub-menu.



Module 12-2: Config AYDIN BIT SYNC MODULE Window

Module 12-3: AYDIN BIT SYNC MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			

Parameter	Unit	Range or Values	Description
Code	User Selection	NRZ-L INRZ-L NRZ-M NRZ-S RNRZ-L IRNRZ-L RNRZ-L(Rev) IRNRZ-L(Rev) BIO-L IBIO-L BIO-M BIO-S DM-M DM-S MDM-M MDM-S	Input data format.
Source	User selection	Input A Input B	Selects which of two inputs is to be processed. The other is ignored.
Bit Rate	Decimal Hz		Expected (nominal) Bit Rate
Center Frequency Restoration			Enables/Disables CFR. On/Off checkbox.
OUTPUT			
code		NRZ-L NRZ-M NRZ-S RNRZ-L BIO-L BIO-M BIO-S	Desired data output code.
INPUT LOOP			
Type		Normal Closed Normal Open Reset Closed Reset Open	Specifies Open/Closed, Normal or Reset Loop. Typically set for Normal Closed, but can be reset or set to open loop for testing.
Bandwidth		0.1 to 1.6%	Specifies allowable bandwidth, in 0.1% resolution. Allows user to lock up on data more quickly depending on the setting required to account for data jitter.
VITERBI			
Viterbi		Off Decoder On Encoder On Enc & Dec On	Choice of Viterbi Off, Encoder On, Decoder On, or Both On
OTHER SETUP			
Device	User Selection	Device #	Selects which PC335 is to be controlled as defined in AUXIO.

Extended Options Window Definition

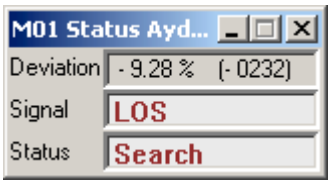
There is no extended options window for the AYDIN BIT SYNC MODULE.

Configuration Notes

There are no configuration notes for the AYDIN BIT SYNC MODULE.

Status Window Definition

When you click the Status button on the AYDIN BIT SYNC MODULE window, a Status AYDIN BIT SYNC window appears, as shown below.



Module 12-4: Status AYDIN BIT SYNC MODULE Window

Module 12-5: AYDIN BIT SYNC MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Deviation	percentage		± Deviation from ideal expressed as a percentage and the actual raw value received form the card.
Signal		Normal LOS	Signal presence indicator.
Status		Search Phase Lock	Correlator status indicator.

Module 13 AYDIN BIT SYNC WITH VITERBI MODULE

MODULE NAME: AydinBitSyncViterbi.dll

Key Functions

- Bit Synchronization and carrier restoration of a serial stream
- Controls Aydin PC 335V Card
- Viterbi decoding

Required Hardware

- The following card must be installed in the Avtec PTP for Windows system:
  - ✓ Aydin PC 335V Card
- Appropriate cables to interface with the card.

Add Module Guidelines

Typically, one instance of the module is used to control each card. The quantity of bit sync cards supported is limited only by the number of available ISA bus slots in the computer. The user must have configured each instance of the card for the AUXIO driver. This is accomplished using the supplied AUXIO_CONFIG utility program, which generates the appropriate system registry information to control the Aydin cards. The cards are numbered sequentially, beginning with 1, as defined in the registry. The order of the cards electrically on the ISA bus is of no consequence.

Required Input Source, Connections, and Events

INPUT SOURCE: All data input is supplied through external cables plugged into the card.

CONNECTIONS: No module connections are supported.

EVENTS: No module events are expected or generated.

Module Functional Description

This module interfaces with the Aydin PC-335V card over the ISA bus, using the AUXIO driver. The user must have configured each card in the system registry using the AUXIO_CONFIG utility provided with the Avtec PTP for Windows software. Only control and status information is placed on the ISA bus; all user data in and data with clock is handled directly by the PC-335V card via external cabling. The card may be controlled by the console GUI or by Remote Interface Library (RIL) calls.

Input Data Format

The PC-335V accepts data in any of the NRZ (L,M,S) or BIØ (L,M,S) formats, see table below for a complete list.

Output Data Format

The user may select the desired output format. See table below.

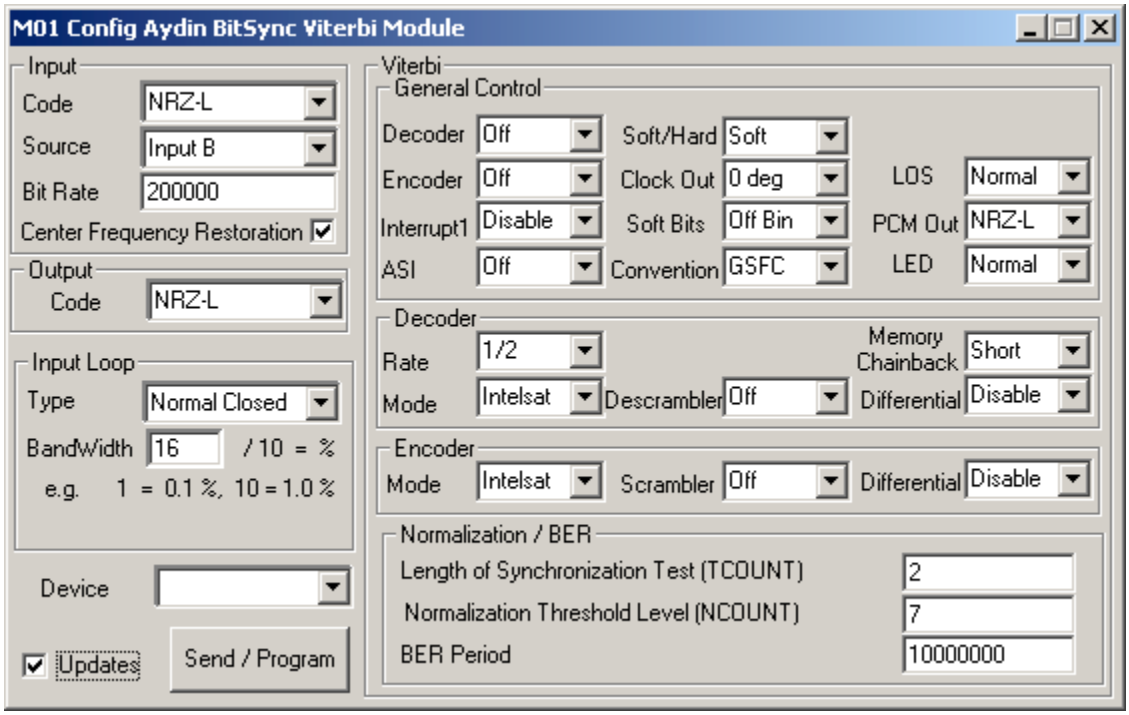
Module Window

When *AydinBitSyncViterbi.dll* is added to a desktop, the Aydin Bit Sync with Viterbi dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 13-1: AYDIN BIT SYNC WITH VITERBI MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AYDIN BIT SYNC WITH VITERBI MODULE Board Parameters sub-menu.



Module 13-2: Config AYDIN BIT SYNC WITH VITERBI MODULE Window

Module 13-3: AYDIN BIT SYNC WITH VITERBI MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			
Code	Selectable Setup State	NRZ-L INRZ-L NRZ-M NRZ-S RNRZ-L IRNRZ-L RNRZ-L(Rev) IRNRZ-L(Rev) BIO-L IBIO-L BIO-M BIO-S DM-M DM-S MDM-M MDM-S	Input data format.
Source	Selectable Setup State	Input A, Input B	Selects which of two inputs is to be processed. The other is ignored.
Bit Rate	Bits per second		Expected (nominal) bit rate.

Parameter	Unit	Range or Values	Description
Center Frequency Restor-ation	Selectable Setup State	Enable Disable	On/Off check box.
VITERBI GENERAL CONTROL			
Decoder		On Off	Turns Viterbi decoder on or off.
Encoder		On Off	Turns Viterbi Encoder on or off.
Interrupt1		Enable Disable	See Aydin manual for details.
ASI		On Off	Alternate Symbol Inversion on/off.
Soft/Hard		Soft, Hard	Soft or Hard decision mode. See Aydin manual for a complete description; depending on the configuration of other parameters, this parameter may or may not be user selectable.
Clock Out		0 deg 180 deg	Output Clock phase; 0 degrees aligns clock and data edges, 180 degrees puts clock leading edge half way into the bit time.
Soft Bits		Off Bin, Sign Mag	Format of soft decision bits; signed or unsigned.
Convention		GSFC, JPL	Expects either GSFC or JPL data conventions. See Aydin manual for a complete description.
LOS		Normal, LEDs On	Causes the light emitting diodes on the card to light during loss of signal condition.
PCM Out		NRZ-L INRZ-L NRZ-M NRZ-S BIO-L IBIO-L BIO-M BIO-S RNRZ-L	Pulse Code Modulation output format selected by user.
LED		Normal Test	Allows user to test the onboard light emitting diodes.
OUTPUT			
Code		NRZ-L NRZ-M NRZ-S BIO-L BIO-M BIO-S RNRZ-L	Desired data output code.
INPUT LOOP			

Parameter	Unit	Range or Values	Description
Type		Normal Closed, Normal Open, Reset Closed, Reset Open	Input Loop type. Typically set for Normal Closed, but can be reset or set to open loop for testing.
Bandwidth		0.1% to 1.6%	Loop Bandwidth. Allows user to lock up on data more quickly depending on the setting required to account for data jitter.
DECODER			
Rate		1/2 1/3	Convolutional decoder rate.
Mode		IntelSat CCIT V.35	See PC-335V manual for a description.
Descram-bler		Off On	See PC-335V manual for a description.
Memory Chainback		Short Full	See PC-335V manual for a description.
Differen-tial		Disable Enable	See PC-335V manual for a description.
ENCODER			
Mode		IntelSat CCIT V.35	See PC-335V manual for a description.
Scrambler		Off On	See PC-335V manual for a description.
Differential		Disable Enable	See PC-335V manual for a description.
NORMALIZATION/BER			
Length of Synchron-ization Test (TCOUNT)			See PC-335V manual for a description.
Normaliza-tion Threshold Level (NCOUNT)			See PC-335V manual for a description.
BER Period			See PC-335V manual for a description.
CARD-LEVEL CONFIG			
Device		Select Device	Selects which card that the module will control.

Extended Options Window Definition

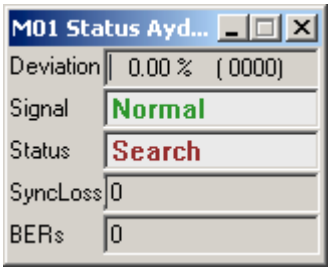
There is no extended options window for the AYDIN BIT SYNC WITH VITERBI MODULE.

Configuration Notes

The numerical order of the cards is defined by the system registry.

Status Window Definition

When you click the Status button on the AYDIN BIT SYNC WITH VITERBI MODULE window, a Status AYDIN BIT SYNC WITH VITERBI window appears, as shown below.



Module 13-4: Status AYDIN BIT SYNC WITH VITERBI MODULE Window

Module 13-5: AYDIN BIT SYNC WITH VITERBI MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Deviation	percentage		± Deviation from ideal expressed as a percentage and the actual raw value received form the card.
Signal		Normal LOS	Signal presence indicator.
Status		Search Phase Lock	Correlator status indicator.
Sync Loss			Count of Sync Losses as reported by the card.
BERs			Bit Errors as reported by the board.

Module 14 AYDIN PSK DEMODULATOR MODULE

MODULE NAME: PSKDemod.dll

Key Functions

- Controls Aydin PSK Demodulator
- Reports status as received from the demodulator

Required Hardware

- The Aydin PSK Demodulator must be installed in the system.

Add Module Guidelines

The operator can use one module per card to be used.

Required Input Source, Connections, and Events

INPUT SOURCE: All data comes from cables plugged into the back of the card.

CONNECTIONS: No module connections are made.

EVENTS: No events are generated or expected.

Module Functional Description

The Aydin PSK Demodulator Module (Aydin PSKDemod) supports the Aydin PSK Demodulator card. The PSK Demodulator module does not have any data inputs or outputs or event outputs and cannot be "connected" to another module. The PSK Demodulator board performs subcarrier detection and synchronous demodulation of an input PSK modulation channel contaminated by wide band noise and embedded within a multiplex of other PSK and FM channel.

Input Data Format

Modulated data.

Output Data Format

Demodulated data.

Module Window

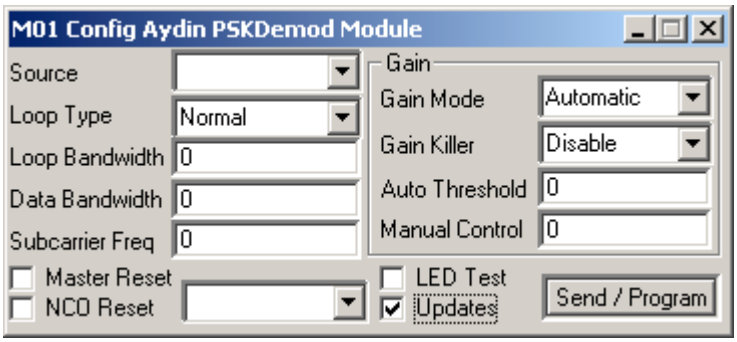
When Aydin *PSKDemod.dll* is added to a desktop, the Aydin PSK Demodulator dialog appears. Left mouse click the Configure button to view current setup or change configuration

parameters. Left mouse click Status to view current module state and data processing statistics.

Module 14-1: AYDIN PSK DEMODULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure AYDIN PSK DEMODULATOR MODULE Board Parameters sub-menu.



Module 14-2: Config AYDIN PSK DEMODULATOR MODULE Window

Module 14-3: AYDIN PSK DEMODULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Source	Selectable Setup	Source 1 Low Z Source 1, Hi Z Source 2, Low Z Source 2, Hi Z	Selects between source 1 or 2, also Low or Hi Z (Impedance)
Loop Type	Selectable Setup	Normal Open	Permits Normal or Open Loop
Loop Bandwidth	Decimal Hz		Specifies loop bandwidth in Hz. Specifies allowable deviation from ideal rate.
Data Bandwidth	Decimal Hz		Specifies data bandwidth in Hz
Subcarrier Freq	Decimal	1000 to 10000000	Specifies expected subcarrier frequency in Hz

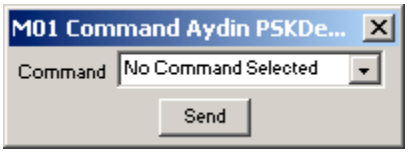
Parameter	Unit	Range or Values	Description
Master Reset	Selectable Setup	Enable Disable	Instructs board to perform a master reset and re-programming
NCO Reset	Selectable Setup	Enable Disable	Instructs board to perform an NCO reset and re-sweep
LED Test	Selectable Setup	Enable Disable	Turns on all LEDs on the PSKDemod board for a visual inspection
GAIN			
Gain Mode	Selectable Setup	Automatic Normal	Choice of Automatic or Manual Gain control
Gain Killer	Selectable Setup	Enable Disable	Enable or Disable
Auto Threshold	Decimal	0 to 255	Automatic Gain Threshold
Manual Control	Decimal	0 to 255	Manual Gain Control Threshold Setting

Extended Options Window Definition

There is no extended options window for the AYDIN PSK DEMODULATOR MODULE.

Configuration Notes

When you click the Command button on the AYDIN PSK DEMODULATOR MODULE window, a Command AYDIN PSK DEMODULATOR window appears, as shown below.



Module 14-4: Command AYDIN PSK DEMODULATOR MODULE Window

Module 14-5: AYDIN PSK DEMODULATOR MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected			No command selected
Master Reset			Initiates a Master Reset of the board, followed by a re-programming
NCO Reset			Initiates a NCO Reset of the board, followed by a reprogramming
Master & NCO Reset			Initiates both a Master and NCO Reset of the board, followed by a reprogramming

Status Window Definition

When you click the Status button on the AYDIN PSK DEMODULATOR MODULE window, a Status AYDIN PSK DEMODULATOR window appears, as shown below.

Module 14-6: Status AYDIN PSK DEMODULATOR MODULE Window

Module 14-7: AYDIN PSK DEMODULATOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Signal Strength	Raw Decimal, Status, Calibrated		Direct raw value readout from the board, with its validity indicator, followed by a calibrated Signal Strength in Volts peak-to-peak
Deviation	Raw Decimal, Status, Calibrated		Direct raw value readout from the board, with its validity indicator, followed by a calibrated deviation read out in LoopBandWidths
Status			Displays Unknown, Subcarrier Sync, Signal LOS, or Overload

Module 15 BCH DECODER MODULE

MODULE NAME: BCHDecoder.dll

Key Functions

- Receives Command Link Transmission Units (CLTUs) from another module
- Optionally performs derandomization of the codeblock data prior to dencoding
- Performs BCH decoding with programmable codeblock size
- Decodes the CLTU into a TC Frame
- Outputs the TC Frame to another module

Required Hardware

- None

Add Module Guidelines

System limitations determine the number of BCH ENCODER MODULES that can be added to a desktop. The total maximum allowable number of modules in a desktop is 32 and this number can not be exceeded. Since this is a software module, the CPU and data rates also determine how many modules can simultaneously process data.

Required Input Source, Connections, and Events

INPUT SOURCE: The BCH Decoder Module accepts CLTUs posted from other modules.

CONNECTIONS: The BCH Decoder Module posts the TC Frames to Port 1.

EVENTS: No events required. No events generated.

Module Functional Description

The BCH Decoder Module implements the inverse of the Coding layer defined in the CCSDS Telecommand Part 1: Channel Service Recommendation (CCSDS 201.0-B-2, November 1995). The Coding layer establishes the reliable, error-controlled data channel through which user telecommand data bits may be transferred. The data are encoded to reduce the effects of noise in the Physical layer channel on the user data. A block code has been chosen to provide this protection. Synchronization for the codeblock and delimiting of the beginning of user data are provided by the Command Link Transmission Unit (CLTU) data structure.

Input Data Format

The input data buffers must contain a single CLTU.

Output Data Format

The output data (Port 1) contains a single TC Frame.

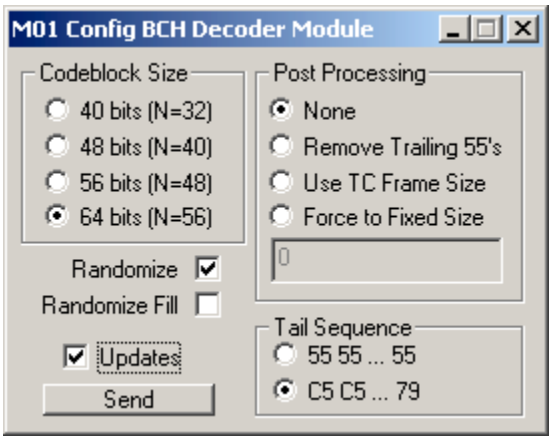
Module Window

When BCHDecoder.dll is added to a desktop, the BCH Decoder dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 15-1: BCH DECODER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure BCH DECODER MODULE Board Parameters sub-menu.



Module 15-2: Config BCH DECODER MODULE Window

Module 15-3: BCH DECODER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description	
Codeblock Size	Bits	40, 48, 56, 64	Size (in bits) of the codeblocks within the CLTU	
Randomize	Check Box	Boolean	Check to use the randomize routine when creating the CLTU	
Post Processing	Radio Button	Radio Selection	The post processing section helps the module determine how to size the data after decoding.	
			None	Decode the CLTU and leave size as is. This is useful if the output is not *actually* a TC Frame

Parameter	Unit	Range or Values	Description	
			Remove Trailing 55's	A risky sizing scheme where the trailing 55's are removed. This may result in lost data if the frame ends in 55
			Use TC Frame Size	Assume the result is a valid TC Frame and use the TC Frame header to determine the size
			Force to Fixed Size	Use this to force the size to a known value. May be used for fixed sized frames only.
Tail Sequence	Radio Buttons	55 55 … 55, C5 C5 … 79	Select the tail sequence to use for the CLTU	

Extended Options Window Definition

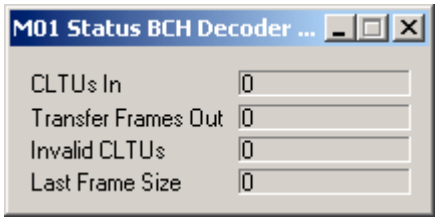
There is an extended options window for the BCH DECODER MODULE MODULE.

Configuration Notes

The BCH Decoder Module is included in the standard module set.

Status Window Definition

When you click the Status button on the BCH DECODER MODULE window, a Status BCH DECODER window appears, as shown below.



Module 15-4: Status BCH DECODER MODULE Window

Module 15-5: BCH DECODER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
CLTUs In	Counter	32 bits	Count of CLTUs accepted from input port
Transfer Frames Out	Counter	32 bits	Count of TC Frames actually posted to Port 1
Invalid CLTUs	Counter	32 bits	Count of CLTUs discarded due to error
Last Frame Size	Bytes	32 bits	Size of last frame sent

Module 16 BCH ENCODER MODULE

MODULE NAME: BCHEncoder.dll

Key Functions

- Receives Telecommand Transfer Frames (TC Frames) from another module (e.g. LEO-T Receiver)
- Optionally performs randomization of the codeblock data prior to encoding
- Performs BCH coding with programmable codeblock size
- Generates a Command Link Transmission Unit (CLTU) by prepending the start sequence and appending the tail sequence to the TC codeblocks (One TC Frame per CLTU)
- Outputs Command Link Transmission Units (CLTUs) to another module (e.g. Serial Output)

Required Hardware

- None

Add Module Guidelines

System limitations determine the number of BCH ENCODER MODULES that can be added to a desktop. The total maximum allowable number of modules in a desktop is 32 and this number can not be exceeded. Since this is a software module, the CPU and data rates also determine how many modules can simultaneously process data.

Required Input Source, Connections, and Events

INPUT SOURCE: The BCH Encoder Module accepts TC Frames posted from other modules.

CONNECTIONS: The BCH Encoder Module posts the CLTUs to Port 1.

EVENTS: No events required. No events generated.

Module Functional Description

The BCH Encoder Module implements the Coding layer defined in the CCSDS Telecommand Part 1: Channel Service Recommendation (CCSDS 201.0-B-2, November 1995). The Coding layer establishes the reliable, error-controlled data channel through which user telecommand data bits may be transferred. The data are encoded to reduce the effects of noise in the Physical layer channel on the user data. A block code has been chosen to provide this protection. Synchronization for the codeblock and delimiting of the beginning of user data are provided by the Command Link Transmission Unit (CLTU) data structure.

Input Data Format

The input data buffers must contain a single TC Frame.

Output Data Format

The output data (Port 1) contains a single CLTU.

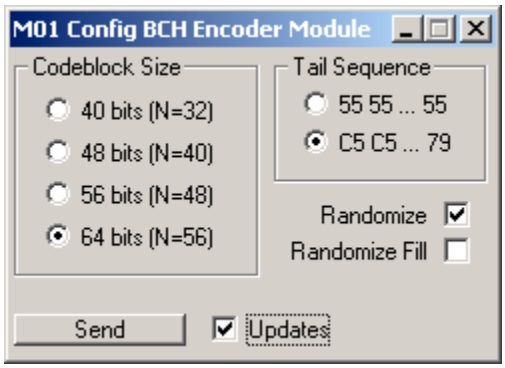
Module Window

When BCHEncoder.dll is added to a desktop, the BCH Encoder dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 16-1: BCH ENCODER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure BCH ENCODER MODULE Board Parameters sub-menu.



Module 16-2: Config BCH ENCODER MODULE Window

Module 16-3: BCH ENCODER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Codeblock Size	Bits	40, 48, 56, 64	Size (in bits) of the codeblocks within the CLTU
Tail Sequence	Radio Buttons	55 55 … 55, C5 C5 … 79	Select the tail sequence to use for the CLTU
Randomize	Check Box	Boolean	Check to use the randomize routine when creating the CLTU

Extended Options Window Definition

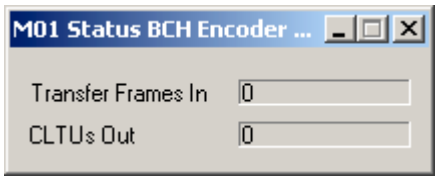
There is no extended options window for the BCH ENCODER MODULE MODULE.

Configuration Notes

The BCH Encoder Module is included in the standard module set. This module is best used in conjunction with the Ames Encapsulator module. The data of the BCH encoder module should be sent to an Ames encapsulator module, which prepends a 555…. (Up to 32 count which equals 16 bytes) sequence. This 555… acquisition sequence is required for a bit sync lock in the spacecraft.

Status Window Definition

When you click the Status button on the BCH ENCODER MODULE window, a Status BCH ENCODER window appears, as shown below.



Module 16-4: Status BCH ENCODER MODULE Window

Module 16-5: CLCW PROCESSOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Transfer Frames In	Counter	32 bits	Count of number of TC Frames accepted from input port
CLTUs Out	Counter	32 bits	Count of number of CLTUs actually posted to Port 1

Module 17 BEST SOURCE SELECT MODULE

MODULE NAME: BestSourceSelect.dll

Key Functions

- Accepts multiple, identical, input sources and outputs only the best source
- Provides user-defined sample period and error thresholds: both values determine current best source
- Switches output based on current best source during data flow

Required Hardware

- No special hardware required.

Add Module Guidelines

Do not exceed 32 modules per desktop system-defined limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Accepts multiple input sources (any other module).

CONNECTIONS: There is one output stream: outputs to port one.

EVENTS: Events are not generated.

Module Functional Description

The Best Source Select (BSS) Module is designed for applications where the user has multiple, identical data streams and needs to compare each stream so that only the "best" stream is accepted and output from the module.

The BSS Module accepts multiple input streams, determines which stream is "best," and than outputs only the selected (best) stream. To determine which input is best, the module maintains an input frame count for each source during a predefined Sample Period and uses the user-defined Error Threshold number. At the end of a Sample Period (sample period length and start defined by user), the BSS Module compares the input frame count of each source stream. If a source stream has a frame count that is more than the other sources, and the amount more is equal to or greater than the Error Threshold value, then the source is considered the "best" and that is the stream that is output from the module. At start, the BSS outputs a "preferred" data source that the user defines, and then switches (or maintains) based on frame count comparisons at the end of each sample period.

At the end of each sample period, all input source frame counts goes back to zero, i.e., frames counts are not cumulative from one sample to the next.

Input Data Format

This module expects frame input, but does not process data contents, so any fixed length buffer size can be input. The data should be "pre-qualified" prior to input to this module. This means that it has been frame synchronized (and Reed-Solomon decoded if applicable) so that only good quality data is input to the module. Since the module chooses the best source based on frame counts, the logic is most effective when the counts are made based on good data.

Output Data Format

This module outputs the selected data source. Only one source stream is output regardless of the number of data streams input. Selected source stream can change during data flow. Frames (or buffers of data) are output exactly as received.

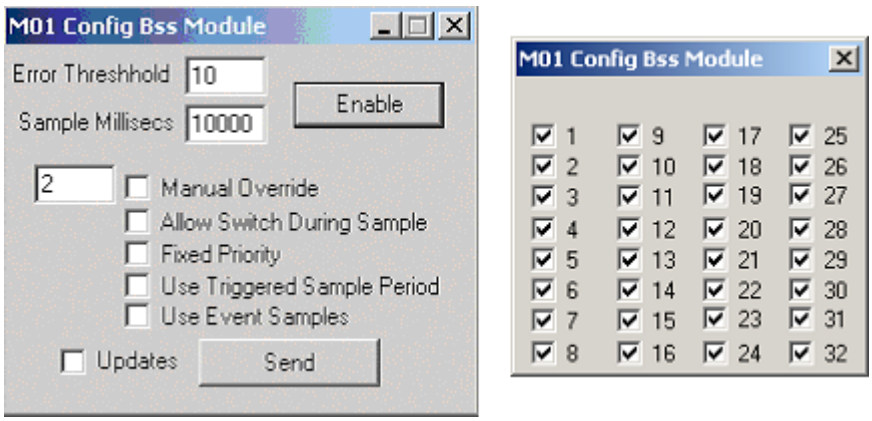
Module Window

When BestSourceSelect.dll is added to a desktop, the Best Source Select dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics. This window includes a Command button, which also brings up a window at left mouse click.

Module 17-1: BEST SOURCE SELECT MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The figure on the right appears when you click the Enable button. The table that follows defines the fields in the Configure BEST SOURCE SELECT MODULE Board Parameters sub-menu.



Module 17-2: Config BEST SOURCE SELECT MODULE Window

Module 17-3: BEST SOURCE SELECT MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
ENABLE	Access to Another Window	Enable/disable input from source modules 1 to 32	Allows the operator to override the "Edit Modules Connection" information at the input of BSS. Allows a source to be turned off on-the-fly. When this button is selected, it brings up an "Config BSS Module" window. This window is a list of 32 possible inputs. The number represents the source module number. If a check precedes the number, it means that the BSS can receive data from that module, but it won't actually receive data from that module unless the connect modules window is set up that way and the source module exists. To disable a source that is setup in the "Edit Modules Connection" window, remove the check from the box and issue a "Send." This overrides the connection made through the "Edit Modules Connection" window.
Error Threshold	frames		Defines frame count value for switching from one stream to another: this field defines the frame count that must be meet before the logic switches to a new input source stream. This field defines the number of frames ahead a stream must be to qualify for switching. At sample time, if the module receives more frames from one source than the others, and the number more is greater than or equal to the Error Threshold, then the logic switches to the new stream.
Sample Millisecs	Millli-seconds		Defines sample period in milliseconds. A sample period is the amount of time that input counts from each source are maintained, and then at the end of the sample, the number of frames received from each stream are compared.
Manual Override and preferred source definition **text entry box**	Selectable Setup	Enable Disable	The text entry box that precedes the Manual Override check box defines the "preferred" input stream. The preferred input stream is the input source (by module number) that is used at start of data processing. The preferred input stream is also used for manual override check box and Switch Now command. The Manual Override checkbox allows the user to manually select an output stream. If manual override is enabled, it stays on the preferred stream, which is identified by the entry in the text box, regardless of sample results. If disabled, then the defined sample decision logic is followed. During a data flow, if the check is removed and a SEND issued, then the decision logic is allowed to start to work.
Allow Switch During Sample	Selectable Setup	Enable Disable	Defines when the module switches to a new source: immediately or at end of sample period. If enabled, then the module immediately switches to the new input source as soon as the error threshold is met. Insert check to enable. If disabled, the module must wait until end of sample period to switch and the decision is based on the statistics at end of sample period.

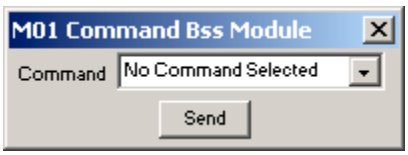
Parameter	Unit	Range or Values	Description
Fixed Priority	Selectable Setup	Enable Disable	Defines the logic used to select the input source when multiple input streams have the same quality (same number of frames input). Decisions are made based on source module numbers. If Enabled (check inserted), the logic maintains a fixed priority – for multiple streams of same quality, always makes decision to use source with lowest module number. If Disabled, the logic is rotational – for multiple streams of same quality, it makes decision based on current source module number. Starts with the current input stream and incrementing up by module number, it goes round robin based on module numbers, and chooses the input source to be next module from the current module
Use Triggered Sample Period	Selectable Setup	Enable Disable	Defines the start of the first sample period. Defines a one-time event at start of desktop enable. If disabled, the sample window is free running. The first sample period starts at receipt of data, and succeeding samples are based on setup sample period. If enabled, it synchronizes the Sample Period timer to first detected error. This means that the first sample time starts as soon as one stream is two frames ahead. From that point, succeeding samples are based on setup sample period.
Use Event Samples	Selectable Setup	Enable Disable	Functionality is TBS. Leave in a disabled state.

Extended Options Window Definition

There is no extended options window for the BEST SOURCE SELECT MODULE.

Configuration Notes

When you click the Command button on the BEST SOURCE SELECT MODULE window, a Command BEST SOURCE SELECT window appears, as shown below.



Module 17-4: Command BEST SOURCE SELECT MODULE Window

Module 17-5: BEST SOURCE SELECT MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	N/A	Selected/ Not Selected	Disables the command function. Issues no command if SEND issued.

Parameter	Unit	Range or Values	Description
Switch Now	N/A	Selected/ Not Selected	When Send is issued, this command forces the BSS Module to switch streams from the current source to the preferred source . The stream that the BSS module will switch to is the stream number entered in the Config BSS Module window in the edit box beside the Manual Override checkbox. Switches to the manual best source for the duration of the current sample period.

Status Window Definition

When you click the Status button on the BEST SOURCE SELECT MODULE window, a Status BEST SOURCE SELECT window appears, as shown below.

Module 17-6: Status BEST SOURCE SELECT MODULE Window

Module 17-7: BEST SOURCE SELECT MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Active Stream	Module number	1 to 32	Indicates the current stream being output by module number

Module 18 BIT DENSITY CORRECTION MODULE

MODULE NAME: BitDensityCorrection.dll

Key Functions

- Generates bit transitions in a data stream
- Overlays (XORs) a 2047 pattern on a per frame basis to cause/remove bit transitions
- Requires no configuration and provides no status
- Works at transmit and receive to insure data integrity

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide the input.

CONNECTIONS: Outputs to port one, which must be connected to another modules input or the data is lost.

EVENTS: It neither generates an event or expects an event input.

Module Functional Description

The Bit Density Correction Module is designed to guarantee signal transitions in a bit stream. Transitions are made by XORing an incoming data with a 2047 psuedo-random pattern, regardless of frame size. Encoding is done on a per frame basis: the 2047 pattern is reset at the start of each input buffer. Encoding is removed from the frame by passing through a second instance of module on receiving end.

Input Data Format

Accepts buffers input from any module.

Output Data Format

Outputs each buffer input.

Module Window

When BitDensityCorrection.dll is added to a desktop, the Bit Density Correction dialog appears. There are no other windows associated with the BIT DENSITY CORRECTION MODULE: there is neither a status nor configuration window.

Module 18-1: BIT DENSITY CORRECTION MODULE Window

Configure Window Definition

There is no configuration window for the BIT DENSITY CORRECTION MODULE.

Extended Options Window Definition

There is no extended options window for the BIT DENSITY CORRECTION MODULE.

Configuration Notes

There is no configuration notes for the BIT DENSITY CORRECTION MODULE.

Status Window Definition

There is no status window for the BIT DENSITY CORRECTION MODULE.

Module 19 BIT ERROR RATE TESTER MODULE

MODULE NAME: BitErrorRateTester.dll

Key Functions

- Transmits 2047 psuedo-random pattern (NASA standard)
- Receives and verifies 2047 psuedo-random pattern
- Maintains detailed received data and error statistics
- Allows user to inject errors in transmitted data
- Programmable output buffer length (event driven from another module)

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Receives data from any other module.

CONNECTIONS: Outputs data to port one, which must be connected to another module's input or the data is lost.

EVENTS: Requires an event input to determine the transfer buffer length for data output on port one.

Module Functional Description

The BIT ERROR RATE TESTER (BERT) MODULE provides the capability to perform end-to-end system testing using pseudo-random number sequences. The BERT MODULE provides two separate functions: BERT generator and BERT receiver. The two functions can be used independently or together: together they provide the end-to-end testing. The test pattern used is IRIG and NASA standard 2047 bit pseudo-random sequence. The receiver can validate this pattern from any data source since the verification is based on the pattern definition. (Verification is NOT based on a comparison between what is sent by the generator and what is received.)

The generator function requires an event input. It sends data out port one when it receives an event; therefore, the event input determines the output frame (buffer) length. The generator also allows the user to insert errors into the pattern at transmission. To do this, use the BERT config window (defined below) that applies only to the generator. The Single option causes one bit error to be inserted in the current frame output. The Continuous option causes one bit

error to be injected in each frame output. For both errors, it is a walking bit error that starts at bit one and the error moves one bit into the frame with each successive frame that has an error inserted.

For the receiver function, the module receives data from any other module. The input data is verified against the 2047 pattern. Input buffer data is evaluated exactly as received, and it is expected to be a contiguous 2047 pattern. (All headers must have been stripped before data is input).

A typical application of the BERT MODULE uses three modules: BERT, Serial Output, and Serial Input. The BERT MODULE data output is connected to the Serial Output module, the Serial Output module's *event* output is tied back to the BERT MODULE and its *data* output is connected to the Serial Input Module (cable or loopback). The Serial Input module's data output is connected to the BERT MODULE input. The BERT MODULE repeatedly generates a 2047 bit pseudo-random sequence and passes it to the Serial Output Module for transmission. The Serial Input Module receives the pseudo-random data from the Serial Output Module and passes it back to the BERT MODULE. The BERT MODULE performs a bit-by-bit comparison between the received data and the 2047 pattern to verify that no bits are in error. The BERT module status indicates how many blocks of PN data have been transmitted and received with and without errors.

Input Data Format

The receiver function of this module is expecting 2047 psuedo-random pattern input. The pattern must be contiguous from one input buffer to the next with no prepended patterns (the module does no data stripping).

Output Data Format

The transmitter function outputs 2047 psuedo-random pattern in buffer size designated by event input. The pattern is contiguous from one buffer to the next.

Module Window

When BitErrorRateTester.dll is added to a desktop, the Bit Error Rate Tester dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 19-1: BIT ERROR RATE TESTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure BIT ERROR RATE TESTER MODULE Board Parameters sub-menu.

Module 19-2: Config BIT ERROR RATE TESTER MODULE Window

Module 19-3: BIT ERROR RATE TESTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Single	command	Enable Disable	Inject a single error into the next frame output. Insert check to cause error.
Continuous	command	Enable Disable	When enabled, injects an error into every frame output. Insert check to enable.

Extended Options Window Definition

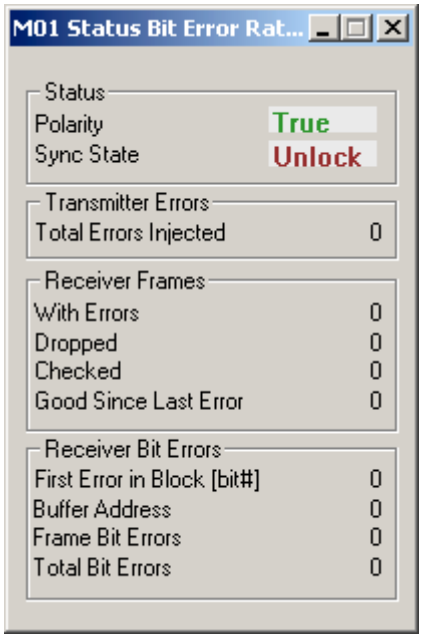
There is no extended options window for the BIT ERROR RATE TESTER MODULE.

Configuration Notes

There are no configuration notes for the BIT ERROR RATE TESTER MODULE.

Status Window Definition

When you click the Status button on the BIT ERROR RATE TESTER MODULE window, a Status BIT ERROR RATE TESTER window appears, as shown below.



Module 19-4: Status BIT ERROR RATE TESTER MODULE Window

Module 19-5: BIT ERROR RATE TESTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
STATUS -- The information in the "status" portion of the window applies to the Receiver function.			
Polarity	Data state	True Inverted	Reports the input data (to the receiver) status. Data Polarity is either True or Inverted. To determine an inverted state, it looks for an inversion in the 2047 pattern
Sync State	Data state	Lock Unlock	Reports if the receiver has locked onto the 2047 pattern. Provides the synchronizer state -- Lock indicates that it sees the pattern
TRANSMITTER ERRORS			
Total Errors Injected	errors	Incrementing count	Reports the Transmitter configuration. It displays the Injected Error Count-- the number of errors that the user inserted with the configuration window.
RECEIVER FRAMES-- note that whatever buffer size is input is a "frame"			
With Errors	frames	Incrementing count	Reports the number of frames received with errors-- any number of errors in a single, continuous buffer is reported as one frame error
Dropped	frames	Incrementing count	Reports the number of frames dropped in the synchronization process-- if the number of errors per frame exceeds 52 bits, that frame is dropped and it synchronizes to the following frame
Checked	frames	Incrementing count	Reports the number of frames checked-- this is the total number of frames received since receiver first received sync-- includes dropped frames
Good Since Last Error	frames	Incrementing count	Reports the number of good frames received since last error-- this is the number of frames received since the last frame with an error in it (any number of errors) was detected

Parameter	Unit	Range or Values	Description
RECEIVER BIT ERRORS-- APPLIES TO THE LAST FRAME THAT HAD AN ERROR			
First Error in Block (bit#)	bits	0 to buffer size	Displays the bit number of the first bit error-- bit number starting from bit one-- the location of the first bit error-- assists in finding stuck bits
Buffer Address	address	N/A	Address of bad buffer-- Address of the Avtec PTP for Windows buffer for that frame
Frame Bit Errors	bits	1 to 52	Total number of bit errors in that frame-- stops counting after 52
Total Bit Errors	bits	Reflects data input	Total detected bit errors for all received data (note that bit error counting stops after 52 for each frame)

Module 20 CCSDS BIT STREAM PROCESSOR MODULE

MODULE NAME: CCSDSBitStreamProcessor.dll

Key Functions

- Perform CCSDS AOS Bitstream Service
- Provides configurable VCDU and Insert Zone lengths
- Performs Bitstream Service on every frame input
- Outputs Bitstream field stripped from frames
- Deletes Insert Zone from VCDU prior to bit stream output
- Optionally includes frame header in output bit stream

Required Hardware

- No additional hardware is required for this module.

Add Module Guidelines

System limitations determine the number of CCSDS BIT STREAM PROCESSOR MODULES that can be added to a desktop. The total maximum allowable number of modules in a desktop is 32 and this number cannot be exceeded. Since this is a software module, the CPU determines how many modules can simultaneously process data.

Required Input Source, Connections, and Events

INPUT SOURCE: The source to this module MUST be the Virtual Channel Processor Version 2 Module.

CONNECTIONS: An input connection must be made from the Virtual Channel Processor Version 2 Module. Since the VC processor can sort by frame VCID, then be certain to connect the CCSDS BIT STREAM PROCESSOR MODULE input to the correct VC Processor output port. Unless an output connection is made from CCSDS BIT STREAM PROCESSOR MODULE port one, then data is dropped although statistics are still maintained and reported to the status window.

EVENTS: This module neither requires an event input nor generates events.

Module Functional Description

This module performs CCSDS AOS Bitstream Service based on frame length and user-configurable VCDU and Insert Zone lengths. The CCSDS BIT STREAM PROCESSOR MODULE is a throughput device. It receives frame input and outputs a bit stream of data.

Unless the "Enable Header Transmission" option is enable, this module strips the frame header, a number of bytes immediately following the header (user-configurable insert zone

length, which can be set to zero), Reed-Solomon symbols, and fill data. The remaining bits of data are output. If the "Enable Header Transmission" option is enabled, then the frame header is output before each bit stream of data is removed from a frame. The insert zone is truncated and whatever is not marked as user data is discarded.

The CCSDS BIT STREAM PROCESSOR MODULE can be used to recreate original spacecraft-generated data that was packaged into frames.

Input Data Format

Input data is CCSDS Version 2 frames.

Output Data Format

Output data is a bit stream of data that reflects the stripping criteria set up in the module configuration window.

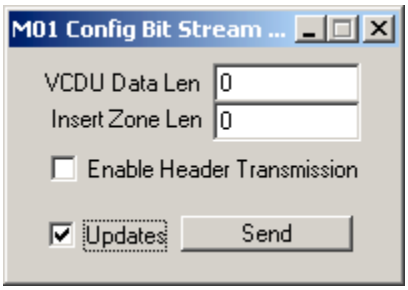
Module Window

When *CCSDSBitStreamProcessor.dll* is added to a desktop, the CCSDS Bit Stream Processor dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 20-1: CCSDS BIT STREAM PROCESSOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CCSDS BIT STREAM PROCESSOR MODULE Board Parameters sub-menu.



Module 20-2: Config CCSDS BIT STREAM PROCESSOR MODULE Window

Module 20-3: CCSDS BIT STREAM PROCESSOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
VCDU Data Len	Bytes	0 to VCDU length	Length of the VCDU. This is the frame length minus the transfer frame header, user insert zone, RS symbols, CLCW, and CRC.
Insert Zone Len	Bytes	0 to VCDU length	This is the length of the insert zone that will be deleted from every frame prior to bit stream output. The insert zone is defined to start immediately following the frame header and continue until "Insert Zone Len."
Enable Header Trans-mission	Selectable Setup State	Enable Disable	This is a check box setup: remove the check to disable and insert check to enable. If this field is enabled, then the frame header is output before each bit stream of data is removed from a frame. The insert zone is truncated and whatever is not marked as user data is discarded.

Extended Options Window Definition

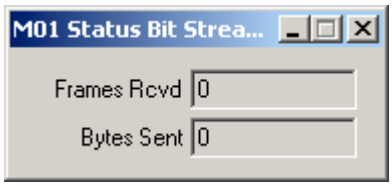
There is no extended options window for the CCSDS BIT STREAM PROCESSOR MODULE.

Configuration Notes

There are no configuration notes for this module.

Status Window Definition

When you click the Status button on the CCSDS BIT STREAM PROCESSOR window, a Status BIT STREAM PROCESSOR window appears, as shown below.



Module 20-4: Status CCSDS BIT STREAM PROCESSOR MODULE Window

Module 20-5: CCSDS BIT STREAM PROCESSOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Frames Rcvd	Frames	Increments by one with each input frame	This is the count of the number of frames input to this module.
Bytes Sent	bytes	bytes output from module	This is the total number of bytes output from the module after stripping is performed on each frame (Bitstream Service). It is total number of bytes output from total frames input. Stripped bytes are based on the configured VCDU data length and insert zone, and frame length.

Module 21 CCSDS VIRTUAL CHANNEL PROCESSOR MODULE

MODULE NAME: VirtualChannelProcessor.dll

Key Functions

- Provides software Reed-Solomon checking and correction: interleaves 1 through 8 and codeword virtual fill are supported
- Provides 16-bit CRC checking with programmable CRC location
- Optionally strips synchronization pattern and/or Reed-Solomon symbols from frames prior to output
- Filters and/or Sorts frame output based on VCID
- Outputs frames to up to four ports (up to 32 output ports are available via remote interface)
- Maintains frame-level quality statistics for each output buffer (frame) that can be used by succeeding modules (required by many encapsulation modules)

Required Hardware

- Requires no special hardware be installed in the Avtec PTP for Windows system.

Add Module Guidelines

As many modules as the system supports can be added (32 maximum per desktop).

Required Input Source, Connections, and Events

INPUT SOURCE: Any CCSDS frame aligned input source.

CONNECTIONS: Outputs to ports as defined by configuration. Ports available for output are 1 through 32. Any port setups after 4 require remote configuration for connections. However, any output port can be set up in the configuration window.

EVENTS: This module neither requires an event input nor generates events.

Module Functional Description

The CCSDS VIRTUAL CHANNEL PROCESSOR MODULE (VCP) receives frame data with quality annotation from another module. The VCP provides a software CRC decoder and a software Reed-Solomon decoder. It can also filter frames based on virtual channel ID and data quality and outputs accepted frames to downstream modules. The VCP supports both Version 1 Transfer Frames and Version 2 Channel Access Data Units. The VCP maintains per virtual channel counts of frames received, frames with RS errors, frames with uncorrectable RS errors, frames with CRC errors, and frames with sequence errors.

The VCP provides multiple data output ports. Each VCP output port can be configured to pass a particular subset of VCDU data. For example, VCP output port 1 could be configured to pass only VCID 0 and VCP output port 2 could be configured to pass VCID 1 and 2.

Input Data Format

The expected input data format is CCSDS Version 1 or Version 2 frames. To provide all available functionality, the input data must be from the Serial Input Module and include extended status (extended status is enabled on the Serial Input Module extended config window).

Output Data Format

Output data is CCSDS Version 1 or Version 2 frames. At output, the input data stream can be filtered or sorted to separate ports by VCID value. Output frame format may be one of the following:

- Complete CCSDS frame including synchronization pattern and Reed-Solomon symbols
- CCSDS frame with synchronization pattern stripped
- CCSDS frame with Reed-Solomon symbols stripped
- CCSDS frame with synchronization pattern and Reed-Solomon symbols stripped

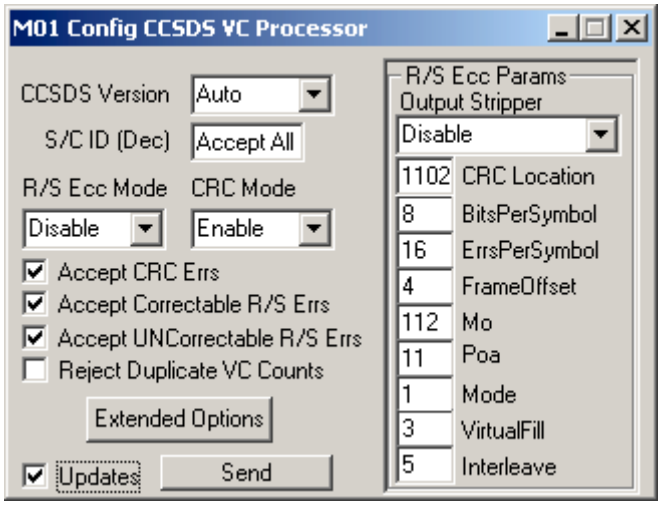
Module Window

When *VirtualChannelProcessor.dll* is added to a desktop, the CCSDS Virtual Channel Processor dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 21-1: CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Board Parameters sub-menu.



Module 21-2: Config CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Window

Module 21-3: CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Configure Window Parameters

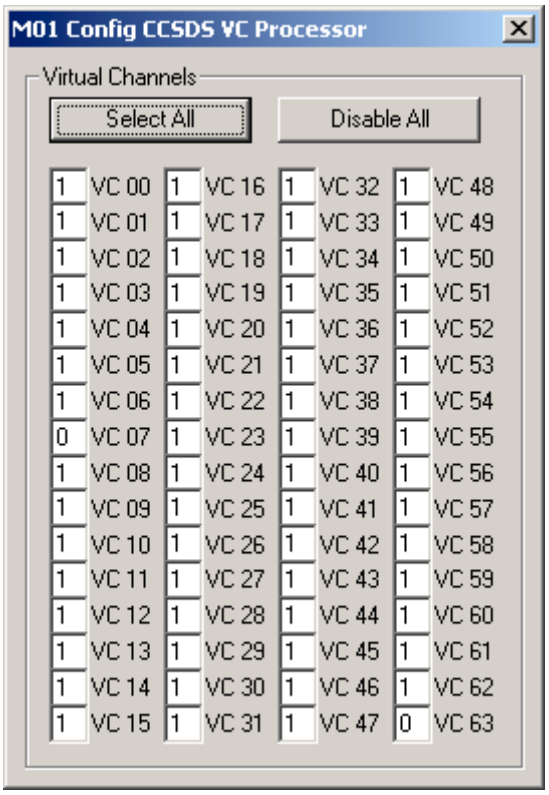
Parameter	Unit	Range or Values	Description
CCSDS Version	Selectable Setup State	1 only 2 only Auto	Defines the expected and accepted frame version. Provides filtering based on version, which means the module won't process the frame unless it is the expected version. Options are as follows: 1) Auto - detect version and accept either: 2) Version 1 - accept only Version 1 3) Version 2 - accept only Version 2
S/C ID (Dec)	decimal		Defines the accepted frame Spacecraft ID (SCID) value. The SCID value read from each input frame header is compared to the configuration value, and only those frames with matching SCID values are processed. Enter "Accept All" to accept any SCID. Otherwise, enter the expected SCID value.
R/S Ecc Mode	Selectable Setup State	Disable Hard Check Correct Soft Check	Defines the type of Reed-Solomon processing performed. Controls the software Reed-Solomon decoder or sets module up to report previous Reed-Solomon statistics recorded by hardware (Serial Input Module). Setup options are defined as follows: 1) Disable - disables the software RS decoder. No RS reporting performed. 2) Hard Check - expects that the Serial Input module filled in the RS status. The input RS status is reported in the status window. (no software RS checking performed). 3) Correct - enables the software RS decoder for error detection and correction. 4) Soft Check- enables the software RS decoder for error detection: enables software check and status report with no correction (slower than correction)

Parameter	Unit	Range or Values	Description
CRC Mode	Selectable Setup State	Disable, Enable, Pre Filter R/S	Controls the Software CRC Decoder: 1) Disable – disables software CRC checking 2) Enable – enables software CRC. Enables 16-bit CRC checking. 3) Pre Filter R/S-Applies to when both RS and CRC checking are enabled. If software RS enabled, CRC checking also pre-filters R/S processing: if CRC valid, then Reed-Solomon processing NOT performed; if CRC not valid, Reed-Solomon processing IS performed.
Accept CRC Errs	Selectable Setup State	Enable Disable	Controls the filtering of frames with CRC errors. Checked, frames with CRC errors are accepted. Unchecked, frames with CRC errors are discarded.
Accept Correctable R/S Errs	Selectable Setup State	Enable Disable	Controls the filtering of frames with correctable RS errors. Checked, frames with correctable RS errors are accepted. Unchecked, frames with correctable RS errors are discarded.
Accept UNCorrect-able R/S Errs	Selectable Setup State	Enable Disable	Controls the filtering of frames with uncorrectable RS errors. Checked, frames with uncorrectable RS errors are accepted. Unchecked, frames with uncorrectable RS errors are discarded.
Reject Duplicate VC counts	Selectable Setup State	Enable Disable	Controls the filtering of frames with duplicate VC sequence counts. Checked, discard frames with duplicate VC sequence counts. Unchecked, accept frames with duplicate VC sequence counts.
R/S ECC PARAMS			
Output Stripper	Selectable Setup State	Disable Sync Only Symbols Only Sync&Sym	Defines the output data format. The selection made in this field identifies what parts of a frame will be stripped (removed) from each frame prior to output from the module. 1) Disable- nothing is stripped from each frame 2) Sync Only- synchronization pattern is stripped from each frame (defined as the CCSDS-standard of first four bytes) 3) Symbols Only- Reed-Solomon symbols are stripped from each frame- symbol is defined as everything after the CRC. RS symbols are 32 bytes per interleave depth 4) Sync and Symbols - both synchronization pattern and Reed-Solomon symbols are stripped from each frame
CRC Location	bytes	0 to frame length	Specifies CRC location. Allows the operator to define a location other than the standard. This value must be supplied if CRC checking is enabled.
BitsPer Symbol	bits		Specifies Bits Per Symbol for the RS code
ErrsPer Symbol			Specifies Errors Per Symbol for the RS code
Frame Offset	bytes		Specifies Frame Offset for the RS code block. This is the number of bytes that is skipped at the beginning of each frame when performing RS processing. Set to 4 to skip the attached sync marker.

Parameter	Unit	Range or Values	Description
Mo	N/A	N/A	Fixed Value; Specifies Mo
Poa	N/A	N/A	Fixed Value; Specifies Poa
Mode	N/A	N/A	Fixed Value; Specifies Mode
VirtualFill	bytes	Must match input frame format	Must be defined if RS processing enabled. Specifies Virtual Fill per codeword (fill is 255 - actual codeword length)
Interleave	interleave	1 to 8 Must match input frame format	Must be defined if RS processing enabled. Specifies Interleave depth from 1 to 8

Extended Options Window Definition

Extended options window for the CCSDS Virtual Channel Processor Module.



Module 21-4: Extended Options CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Window

Module 21-5: CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Extended Options Window Parameters

Parameter	Unit	Range or Values	Description

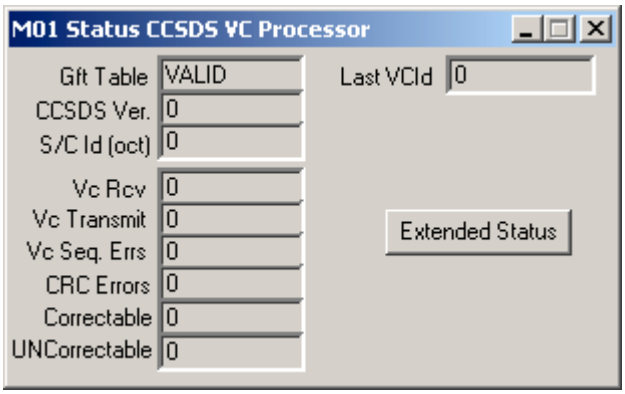
Parameter	Unit	Range or Values	Description
The setup on this window controls the filtering and routing of frames out of the VC Processor. The possible frame VCIDs are listed, and before each VCID value is an entry box. The entry box identifies the port on which each frame with that VCID is output. To filter frames based on VCID, the VCID to be filtered is set to output on port 0.			
Select All	N/A	Command issued	This is a one-step configuration command to set all VCIDs to be output on port 1. No frames are filtered unless the individual VCIDs are edited.
Disable All	N/A	Command issued	This is a one-step configuration command that disables all VC output (sets all data port values to 0).
VC 00 to VC 63	VCID value	0 to 63	Represents all available VCID values and selects which module data port on which each VC is to be output. 0 indicates that the data should be discarded.

Configuration Notes

There are no configuration notes for the CCSDS VIRTUAL CHANNEL PROCESSOR MODULE.

Status Window Definition

When you click the Status button on the CCSDS VIRTUAL CHANNEL PROCESSOR MODULE window, a Status CCSDS VIRTUAL CHANNEL PROCESSOR window appears, as shown below.

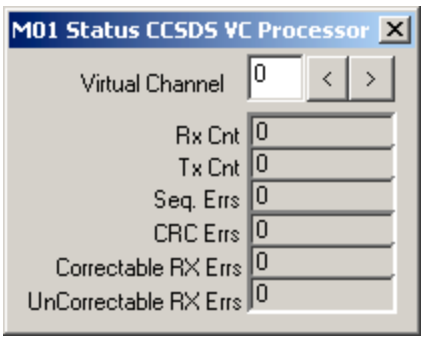


Module 21-6: Status CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Window

Module 21-7: CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Gft Table	State info	Valid Not Valid	Valid/Not Valid Not Valid indicates that a file is missing from the Avtec PTP for Windows directory.
CCSDS Ver.	version	0 or 1	Displays the detected Transfer Frame Version number
S/C Id (oct)	octal	Detected values	Displays the detected Spacecraft ID

Parameter	Unit	Range or Values	Description
Vc Rcv	frames	increments by one for each frame input	Count of total transfer frames received
Vc Transmit	frames	increments by one for each frame output	Count of total transfer frames accepted and transmitted
Vc Seq. Errs	frames	increments by one for each frame with errors	Count of total number of frames with VC sequence errors
CRC Errors	frames	increments by one for each frame with errors	Count of total number of frames with CRC errors
Correctable	frames	increments by one for each frame with correctable errors	Count of total number of frames with correctable RS errors
UNCorrectable	frames	increments by one for each frame with Uncorrectable errors	Count of total number of frames with uncorrectable RS errors
Last VCID	VCID	Detected values	Displays the VCID of the last frame received



Module 21-8: Extended Status CCSDS VIRTUAL CHANNEL PROCESSOR MODULE Window

Module 21-9: CCSDS VIRTUAL CHANNEL PROCESSOR MODULE VCID Extended Status Window Parameters

Parameter	Unit	Range or Values	Description
Virtual Channel	VCID	All detected VCID values	This field allows the operator to control the status display information. All VCID values that have been detected by the module are available in the selection list. To view the statistics for a specific VCID, select the VCID from the list. Once the selection is made, all other information on the Extended Status window changes to reflect the chosen VCID.

Parameter	Unit	Range or Values	Description
Rx Cnt	frames	Incrementing count	Number of transfer frames received for the selected VC
Tx Cnt	frames	Incrementing count	Number of transfer frames accepted and transmitted for the selected VC
Seq. Errs	frames	Incrementing count	Sequence Error Count for the selected VC
CRC Errs	frames	Incrementing count	Number of frames with CRC Errors for the selected VC
Correctable RX Errs	frames	Incrementing count	Correctable RS Errors for the selected VC
UnCorrectable RX Errs	frames	Incrementing count	Uncorrectable RS Errors for the selected VC

Module 22 CCSDS VIRTUAL CHANNEL PROCESSOR VERSION 2 MODULE

MODULE NAME: VirtualChannelProcessorV2.dll

Key Functions

- Provides software Reed-Solomon checking and correction: interleaves 1 through 8 and codeword virtual fill are supported
- Provides 16-bit CRC checking with programmable CRC location
- Filters frames based on quality
- Optionally strips synchronization pattern and/or Reed-Solomon symbols from frames prior to output
- Provides all functions provided by Virtual Channel Processor module and includes several enhancements
- Monitors and reports Master Channel sequence count errors and missing frame counts that are not available with Virtual Channel Processor module
- Filters and/or Sorts frame output based on VCID
- Outputs frames to up to four ports (up to 32 output ports are available via remote interface)
- Maintains and reports frame quality statistics to status window
- Maintains frame-level quality statistics for each output buffer (frame) that can be used by succeeding modules (required by many encapsulation modules)
- Requires significantly different Remote Interface from Virtual Channel Processor module
- Reports the number of missing frames for each frame VCID (based on VCID sequence count gaps), which is a statistic that is not available with Virtual Channel Processor Module.

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of CCSDS VC PROCESSOR VER2 MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Any CCSDS frame aligned input source.

CONNECTIONS: Outputs to ports as defined by configuration. Ports available for output are 1 through 32. Any port setups after 4 require remote configuration for connections. However, any output port can be set up in the configuration window.

EVENTS: This module neither requires an event input nor generates events.

Module Functional Description

The CCSDS VC PROCESSOR VER2 MODULE receives frame data with quality annotation from another module. The VCP provides a software CRC decoder and a software Reed-Solomon decoder. It can also filter frames based on virtual channel ID and data quality and outputs accepted frames to downstream modules. The CCSDS VC PROCESSOR VER2 MODULE supports both Version 1 Transfer Frames and Version 2 Channel Access Data Units. The module maintains per virtual channel counts of frames received, frames with RS errors, frames with uncorrectable RS errors, frames with CRC errors, and frames with sequence errors.

The CCSDS VC PROCESSOR VER2 MODULE provides multiple data output ports. Each VCP output port can be configured to pass a particular subset of VCDU data. For example, VCP output port 1 could be configured to pass only VCID 0 and VCP output port 2 could be configured to pass VCID 1 and 2.

The CCSDS VC PROCESSOR VER2 MODULE has different data and control structures from the Virtual Channel Processor. This means that the RIL interface is radically different between the two modules.

Input Data Format

The expected input data format is CCSDS Version 1 or Version 2 frames. To provide all available functionality, the input data must be from the Serial Input Module and include extended status (extended status is enabled on the Serial Input Module extended config window).

Output Data Format

Output data is CCSDS Version 1 or Version 2 frames. At output, the input data stream can be filtered or sorted to separate ports by VCID value. Output frame format may be one of the following:

- Complete CCSDS frame including synchronization pattern and Reed-Solomon symbols
- CCSDS frame with synchronization pattern stripped
- CCSDS frame with Reed-Solomon symbols stripped
- CCSDS frame with synchronization pattern and Reed-Solomon symbols stripped

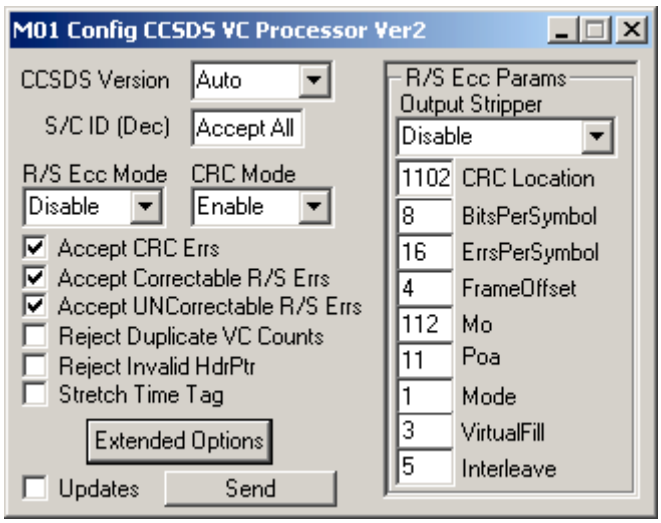
Module Window

When *VirtualChannelProcessorV2.dll* is added to a desktop, the CCSDS VC Processor Ver2 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 22-1: CCSDS VC PROCESSOR VER2 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CCSDS VC PROCESSOR VER2 MODULE Board Parameters sub-menu.



Module 22-2: Config CCSDS VC PROCESSOR VER2 MODULE Window

Module 22-3: CCSDS VC PROCESSOR VER2 MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
CCSDS Version	Version #-Selectable Setup State	1 only 2 only Auto	Defines the expected and accepted frame version. Provides filtering based on version, which means the module won't process the frame unless it is the expected version. Options are as follows: 1) Auto - detect version and accept either: 2) Version 1 - accept only Version 1 3) Version 2 - accept only Version 2
S/CID	decimal	Accept All, SCID value	Defines the accepted frame Spacecraft ID (SCID) value. The SCID value read from each input frame header is compared to the configuration value, and only those frames with matching SCID values are processed. Enter "Accept All" to accept any SCID. Otherwise, enter the expected SCID value.

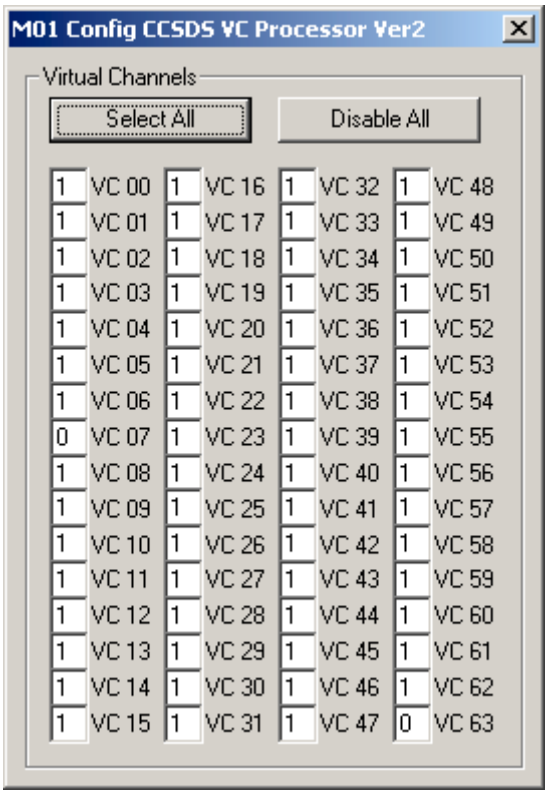
Parameter	Unit	Range or Values	Description
R/S Ecc Mode	Selectable Setup State	Disable Hard Check Correct Soft Check	Defines the type of Reed-Solomon processing performed. Controls the software Reed-Solomon decoder or sets module up to report previous Reed-Solomon statistics recorded by hardware (Serial Input Module). Setup options are defined as follows: 1) Disable - disables the software RS decoder. No RS reporting performed. 2) Hard Check - expects that the Serial Input module filled in the RS status. The input RS status is reported in the status window. (no software RS checking performed). 3) Correct - enables the software RS decoder for error detection and correction. 4) Soft Check- enables the software RS decoder for error detection: enables software check and status report with no correction (slower than correction)
CRC Mode	Selectable Setup State	Disable, Enable, Pre Filter R/S	Controls the Software CRC Decoder: 1) Disable – disables software CRC checking 2) Enable – enables software CRC. Enables 16-bit CRC checking. 3) Pre Filter R/S-Applies to when both RS and CRC checking are enabled. If software RS enabled, CRC checking also pre-filters R/S processing: if CRC valid, then Reed-Solomon processing NOT performed; if CRC not valid, Reed-Solomon processing IS performed.
Accept CRC Errors	Selectable Setup State	Enable Disable Grayed √	Controls the filtering of frames with CRC errors. Checked: frames with CRC errors are accepted. Unchecked: frames with CRC errors are discarded. Grayed √: frames with CRC errors are accepted after first good frame.
Accept Correct-able R/S Errs	Selectable Setup State	Enable Disable Grayed √	Controls the filtering of frames with correctable RS errors. Checked: frames with correctable RS errors are accepted. Unchecked: frames with correctable RS errors are discarded. Grayed √: frames with correctable RS errors are accepted after first good frame.
Accept UN-correctable R/S Errs	Selectable Setup State	Enable Disable Grayed √	Controls the filtering of frames with uncorrectable RS errors. Checked: frames with uncorrectable RS errors are accepted. Unchecked: frames with uncorrectable RS errors are discarded. Grayed √: frames with uncorrectable RS errors are accepted after first good frame.
Reject Duplicate VC Counts	Selectable Setup State	Enable Disable	Controls the filtering of frames with duplicate VC sequence counts. Checked, discard frames with duplicate VC sequence counts. Unchecked, accept frames with duplicate VC sequence counts.

Parameter	Unit	Range or Values	Description
Reject Invalid Hdr/ptr	Selectable Setup State	Enable Disable	When enabled (checked), the module monitors the Packet First Header pointer that is in each frames header. If the pointer is invalid, the frame is filtered (not processed or output) The pointer is invalid if it points outside the transfer frame, which is outside the transfer frame length including Reed-Solomon symbols.
Stretch Time Tag	Selectable Setup State	Enable Disable	Allows adjusting of time stamp to be relative to first bit of data within a transfer frame.
R/S ECC PARAMS			
Output Stripper	Selectable Setup State	Disable Sync Only Symbols Only Sync and Symbols Correctable Symbols Sync and Correctable	Defines the output data format. The selection made in this field identifies what parts of a frame will be stripped (removed) from each frame prior to output from the module. 1) Disable - nothing is stripped from each frame 2) Sync Only - synchronization pattern is stripped from each frame (defined as the CCSDS-standard of first four bytes) 3) Symbols Only - Reed-Solomon symbols are stripped from each frame- symbol is defined as everything after the CRC. RS symbols are 32 bytes per interleave depth 4) Sync and Symbols - both synchronization pattern and Reed-Solomon symbols are stripped from each frame 5) Correctable Symbols – Reed-Solomon symbols are stripped from correctable frames 6) Sync and Correctable – both synchronization pattern and Reed-Solomon symbols are stripped from correctable frames
CRC Location	Bytes	0 to frame length-2	Specifies CRC location. Allows the operator to define a location other than the standard. This value must be supplied if CRC checking is enabled.
Bits Per Symbol	Bits		Specifies Bits Per Symbol for the RS code
Errs Per Symbol	Number		Specifies Errors Per Symbol for the RS code
Frame Offset	bytes	0 to frame length	Specifies Frame Offset for the RS code block. This is the number of bytes that is skipped at the beginning of each frame when performing RS processing. Set to 4 to skip the attached sync marker.
Mo	N/A	N/A	Fixed Value; Specifies Mo
Poa	N/A	N/A	Fixed Value; Specifies Poa
Mode	N/A	N/A	Fixed Value; Specifies Mode
Virtual Fill	bytes	Must match input frame format	Must be defined if RS processing enabled. Specifies Virtual Fill per codeword (fill is 255 - actual codeword length)

Parameter	Unit	Range or Values	Description
Interleave	interleave	1 to 8 Must match input frame format	Must be defined if RS processing enabled. Specifies Interleave depth from 1 to 8

Extended Options Window Definition

When extended options is chosen on the config CCSDS VC PROCESSOR VER2 MODULE window, the figure below illustrates the window that appears.



Module 22-4: Extended Options CCSDS VC PROCESSOR VER2 Window

Module 22-5: CCSDS VC PROCESSOR VER2 Extended Options Parameters

Parameter	Unit	Range or Values	Description
The setup on this window controls the filtering and routing of frames out of the VC Processor. The possible frame VCIDs are listed, and before each VCID value is an entry box. The entry box identifies the port on which each frame with that VCID is output. To filter frames based on VCID, the VCID to be filtered is set to output on port 0.			
Select All	N/A	Command issued	This is a one-step configuration command to set all VCIDs to be output on port 1. No frames are filtered unless the individual VCIDs are edited.

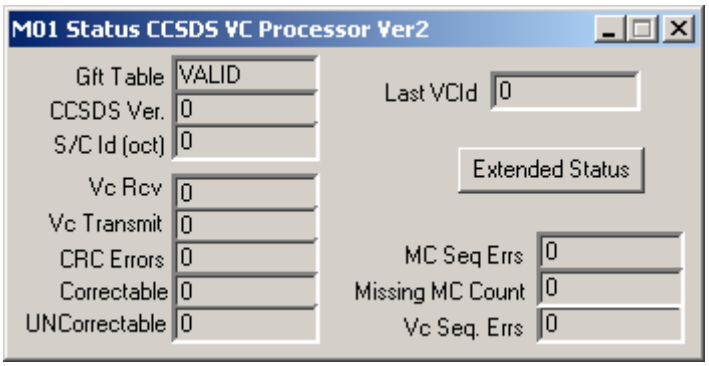
Parameter	Unit	Range or Values	Description
Disable All	N/A	Command issued	This is a one-step configuration command that disables all VC output (sets all data port values to 0).
VC 00 to VC 63	VCID value	0 to 63	Represents all available VCID values and selects which module data port on which each VC is to be output. 0 indicates that the data should be discarded.

Configuration Notes

When generating a desktop from scratch, use this module as the first option (as opposed to the Virtual Channel Processor Module).

Status Window Definition

When you click the Status button on the CCSDS VC PROCESSOR VER2 MODULE window, a Status CCSDS VC PROCESSOR VER2 window appears, as shown below. The figure that follows is the Extended Status window.

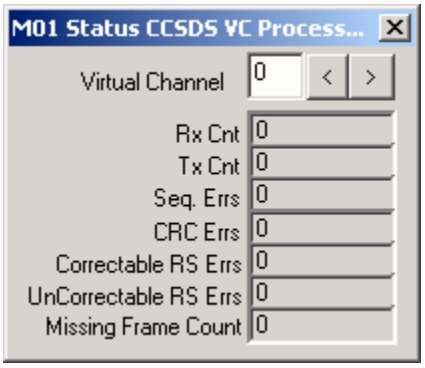


Module 22-6: Status CCSDS VC PROCESSOR VER2 Window

Module 22-7: CCSDS VC PROCESSOR VER2 Status Window Parameters

Parameter	Unit	Range or Values	Description
Gft Table	State info	Valid Not Valid	Valid/Not Valid Not Valid indicates that a file is missing from the PTP directory.
CCSDS ver.	version	0 or 1	Displays the detected Transfer Frame Version number
S/CID	octal	Detected values	Displays the Spacecraft ID detected in the current frame input.
Vc Rcv	frames	increments by one for each frame input	Count of total transfer frames received

Parameter	Unit	Range or Values	Description
Vc Transmit	frames	increments by one for each frame output	Count of total transfer frames accepted and transmitted
CRC Errors	frames	increments by one for each frame with errors	Count of total number of frames with CRC errors
Correctable	frames	increments by one for each frame with correctable errors	Count of total number of frames with correctable RS errors
UN-Correctable	frames	increments by one for each frame with Uncorrect-able errors	Count of total number of frames with uncorrectable RS errors
Last VCID	VCID	Detected values	Displays the VCID of the last frame received.
MC Seq Errs	errors	Incrementing counter	Maintains a master channel sequence error count for Version 1 Transfer frame header. This count is not applicable to Version 2 frames since there is no master channel count in Version 2 frame headers.
Missing MC Count	frames	Incrementing counter	Applies to version 1 frames only. This count is not applicable to Version 2 frames since there is no master channel count in Version 2 frame headers. Maintains a total count of the frames missing in all MC count gaps. For each gap, a maximum of 256 frames can be recorded as missing since the frame MC counter is 8-bits long and rolls over.
Vc Seq Errs	errors	Incrementing counter	Maintains a count of the total number of VCID sequence errors detected in all frames input to module.



Module 22-8: Extended Status CCSDS VC PROCESSOR VER2 Window

Module 22-9: CCSDS VC PROCESSOR VER2 Extended Status Window Parameters

Parameter	Unit	Range or Values	Description
Virtual Channel	VCID	All detected VCID values	This field allows the operator to control the status display information. All VCID values that have been detected by the module are available in the selection list. To view the statistics for a specific VCID, select the VCID from the list. Once the selection is made, all other information on the Extended Status window changes to reflect the chosen VCID.
RxCnt	frames	Incrementing count	Number of transfer frames received for the selected VC
Tx Cnt	frames	Incrementing count	Number of transfer frames accepted and transmitted for the selected VC
Seq Errs	frames	Incrementing count	Sequence Error Count for the selected VC
CRC Errs	frames	Incrementing count	Number of frames with CRC Errors for the selected VC
Correctable RS Errs	frames	Incrementing count	Correctable RS Errors for the selected VC
Un-Correctable RS Errs	frames	Incrementing count	Uncorrectable RS Errors for the selected VC
Missing Frame Count	frames	Incrementing count	Maintains a total count of the number of frames missing in all gaps within that VC. For each gap, there is a maximum number of missing frames that can be accounted for based on the size of the counter in the frame header (counters roll over): 8-bit counter for version 1 24-bit counter for version 2

Module 23 CCSDS VIRTUAL CHANNEL SIMULATOR MODULE

MODULE NAME: CCSDSVirtualChannelSimulator.dll

Key Functions

- Simulates CCSDS Version 1 and Version 2 frame data
- Has several modes of operation that provide control of which VCIDs generated and how data field is packed
- Optionally generates frames filled with CCSDS Version 1 packets
- Accepts another module's input as packet data field

Required Hardware

- The CCSDS VIRTUAL CHANNEL SIMULATOR MODULE requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of CCSDS VIRTUAL CHANNEL SIMULATOR MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Does not require an input source. Can accept input from another module to use as the packet data field for packets put in the data field of generated frames.

CONNECTIONS: The module does not receive input unless a second module is providing source data for the data field of the data unit. For all other data generation there is no data input.

EVENTS: An event is required to trigger the output of the module. It is typically the Serial Output Module that provides the event. The event tells the CCSDS Virtual Channel Simulator what frame length to generate

Module Functional Description

The CCSDS VIRTUAL CHANNEL SIMULATOR MODULE generates Version 1 or Version 2 Transfer Frames. The Virtual Channel Simulator creates Transfer Frames with the 32-bit Attached Sync Marker (1ACFFC1D), a primary header, and fill data in the VCDU data zone. The Virtual Channel Simulator cycles through a preprogrammed virtual channel sequence and generates the sequence count for each frame. The Virtual Channel Simulator also provides software support for CRC encoding and RS encoding.

The Virtual Channel Simulator requires an event input to drive the generation of frames. The Virtual Channel Simulator can be connected to the Serial Output Module to simulate a CCSDS data stream.

Input Data Format

If set up to receive external input, the module uses the input to either fill frame or packet data field zone. In either case, the input data format is irrelevant, except that for filling frames the input data size should equal the frame data field zone.

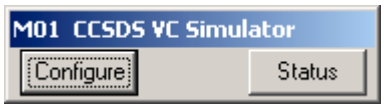
Output Data Format

Module outputs CCSDS Version 1 or Version 2 frames. Data is optionally RS encoded and/or CRC Encoded (16-bit CCITT CRC). Several options for filling packet and frame data zones with BERT patterns that are contiguous within either a VCID or APID are available in setup.

Fill frames are defined as VC 7 for version 1 and VC 63 for version 2

Module Window

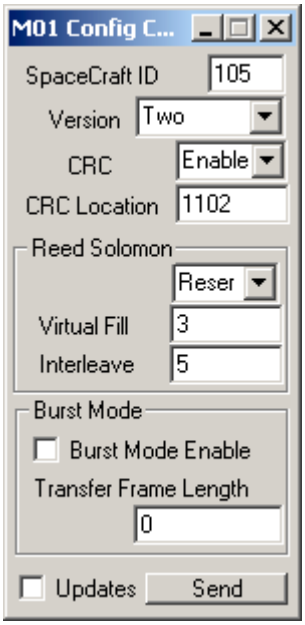
When *CCSDSVirtualChannelSimulator.dll* is added to a desktop, the CCSDS Virtual Channel Simulator dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.



Module 23-1: CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Board Parameters sub-menu.



Module 23-2: Config CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Window

Module 23-3: CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Spacecraft ID	octal		Spacecraft ID that is stamped into the header of every frame output
Version	Selectable Setup	One Two 1 Fill 2 Fill GPB 1 Bert 2 Bert 1 VBert 2 VBert 1 SPBert 2 SPBert 1 MPBert 2 MPBert 1 VExt 2 VExt 1 PExt 2 PExt 1 MPBert 0 2 MPBert 0	Defines the frame type output. A prefix of one or two differentiates between CCSDS version one and two frames. For the MPBert and SPBert, valid APIDs are inserted in the fill channels. The version types are as follows: 1) One: generate CCSDS version 1 frames with VCIDs 0 to 7. VCIDs are generated round robin starting with 0 and ending with 7. VCID 7 is fill and will have a fill pattern of AA. VCID 1 data field pattern is 01; VCID 2 data field pattern is 02; VCID 3 data field pattern is 03; etc. 2) Two: generate CCSDS version 2 frames with VCIDs 0 to 7 and VCID 63. VCIDs are generated round robin starting with 0 and ending with 63. VCID 63 is fill and will have a fill pattern of AA. Same data field patterns as defined for One. 3) 1 Fill: Fill Channel, which generates fill only: version 1 frame, VCID 7 4) 2 Fill: Fill Channel, which generates fill only: version 2 frame, VCID 63 5) GPB: generates version 1 frames of VCID's 0, 1, and 2. VCIDs are generated round robin starting with 0 and ending with 2. 6) 1Bert: Generates version 1 frames. Round robin VCIDs 0 to 7. Everything from the transfer frame header (10 bytes + sync pattern) on is a BERT pattern. The BERT pattern is contiguous from one frame to the next of the same VCID.

Parameter	Unit	Range or Values	Description
			7) 2 Bert: Generates version 2 frames. Round robin VCIDs 0 to 7 and 63. Everything from the transfer frame header (12 bytes + sync pattern) on is BERT pattern. . The BERT pattern is contiguous from one frame to the next of the same VCID 8) 1Vbert: Generates version 1 frames with everything from the VCDU insert zone on filled with a BERT pattern that is contiguous from one frame to the next of the same VCID. VCIDs 0 to 7. 9) 2Vbert: Generates version 2 frames with everything from the VCDU insert zone on filled with a BERT pattern that is contiguous from one frame to the next of the same VCID. VCIDs 0 to 7 and 63. 10) 1SPBert: Generates version 1 frames that are filled with packets. Encapsulates a BERT 2047 pattern into an APID and packs one packet per virtual channel so that the BERT pattern is contiguous from one packet to the next in that VCID. One APID per VCID. The packet length fills the entire frame data field. VCIDs generated are 0 to 7. The APID value packed into each VCID is the [(VCID*32)+1] 11) 2SPBert: Generates version 2 frames that are filled with packets. Encapsulates a BERT 2047 pattern into an APID. One packet per virtual channel so that the BERT pattern is contiguous from one packet to the next in that VCID. One APID per VCID. The packet length fills the entire frame data field. VCIDs generated are 0 to 7 and 63. The APID value packed into each VCID is the [(VCID*32)+1] 12) 1MPBert: Version 1 frames: VCIDs 0 to 7. Generates multiple APIDs per VCID. The number of packets generated per VCID varies with the frame length. The size of the first packet in a VC is 8 bytes. A new APID is generated until the frame data field is filled. Each new APID increases in size by 3 bytes. A new APID is generated until there is 100 bytes or less left in the data field. At this point a "remainder" packet is generated that is the length of the remaining data field. Packet ID is derived from VCID and packet order. A 2047 BERT pattern is contiguous across packets of each APID. 13) 2MPBert: Version 2 frames: VCIDs 0 to 7 and 63. Generates multiple APIDs per VCID. The number of packets generated per VCID varies with the frame length. Refer to the description of packet generation provided in 1MPBert. 14) 1VExt: Version 1 frames. Takes external user data and uses its to fill the frame data field. Any module can provide the external data to the VC Sim. The data unit size provided by the source should equal the data field (VCDU zone). The VCID generated is the number of the module that is feeding the input (data source module). 15) 2VExt: Version 2 frames. Takes external user data and fills the data field. Any module can provide the external data to the VC Sim. The data source size should equal the data field (VCDU

Parameter	Unit	Range or Values	Description
			zone). The VCID generated is the number of the module that is feeding the input (data source module). 16) 1PExt: Version 1 frames. Creates frames with packets in the data field. Uses external input as the packet data field. Takes input from another module and fills the packet data field with that input. The APID value is the source module number. Does wrap packets from frame to frame. Creates only VCID 0 and fill (VCID 7). 17) 2PExt: Version 2 frames. Creates frames with packets in the data field. Uses external input as the packet data field. Takes input from another module and fills the packet data field with that input. The APID value is the source module number. Does wrap packets from frame to frame. Creates only VCID 0 and fill (VCID 7). 18) 1 MPBert 0: Same as 1MPBert but only for Virtual Channel 0. 19) 2 MPBert 0: Same as 2MPBert but only for Virtual Channel 0.
CRC		Enable Disable	Enables or Disable CRC generation and insertion for each frame output. Two CRC bytes are reserved regardless of setup state. By default, the CRC is the last two bytes preceding the RS Symbols or end of frame (whichever comes first). If CRC generation is enabled, then 16 bit CRC is calculated and inserted into the frame. If CRC generation is disabled, the two bytes are filled with zeros.
CRC location	bytes		Must be provided for CRC enabled. Can provide a different location than the standard location defined in CRC description above.
REED-SOLOMON			
RS Mode	Selectable Setup	Reserved Disable Enable	Defines if Reed-Solomon symbols are to be generated for each frame, and if generated how they are inserted. 1) Reserved: using the supplied event length, interleave, and codeword size, outputs a buffer that is the frame length, but the data fills only the length minus RS symbols and CRC (leaves RS and CRC field empty). This can be used in conjunction with the Serial Output Mode set up in overlay mode. 2) Disable: No Reed-Solomon performed. 3) Enable: Calculate Reed-Solomon symbols and insert based on requested frame length prior to module output
Virtual Fill	bytes	0 to (255-32)	Defines the RS virtual fill size, which defines the codeword length given that a codeword length is 255 minus virtual fill
Interleave Depth	interleave	1 to 8	Defines the RS interleave depth
BURST MODE			

Parameter	Unit	Range or Values	Description
Burst Mode Enable	Selectable Setup	Enable Disable	Burst mode overrides standard event mode-- tells it regardless of event frame length, use the Transfer Frame length that is entered. Burst mode also tells it not to generate fill channels
Transfer Frame Length	bytes		Output frame length that will over-ride event provided length when the module is set up to run in Burst Mode.

Extended Options Window Definition

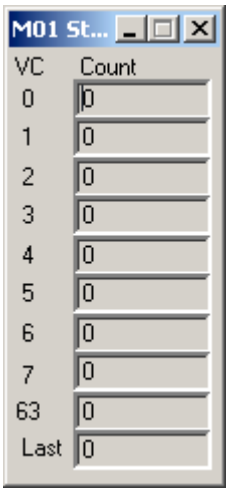
There is no extended options window for the CCSDS VIRTUAL CHANNEL SIMULATOR MODULE.

Configuration Notes

There are no configuration notes for the CCSDS VIRTUAL CHANNEL SIMULATOR MODULE.

Status Window Definition

When you click the Status button on the CCSDS VIRTUAL CHANNEL SIMULATOR MODULE window, a Status CCSDS VIRTUAL CHANNEL SIMULATOR window appears, as shown below.



Module 23-4: Status CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Window

Module 23-5: CCSDS VIRTUAL CHANNEL SIMULATOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
VC 0 to 7 and VC 63	count	Incrementing count	Displays count of frames generated on each virtual channel ID for VCs 0-7 and 63
Last	VCID	0 to 7, and 63	Displays the VC ID for the last frame generated

Module 24 CLCW PROCESSOR MODULE

MODULE NAME: CLCWProcessor.dll

Key Functions

- Extracts CLCW from telemetry stream and posts the CLCW to its output Port 1
- Displays fields of last CLCW
- Optionally only sends changes to the CLCW
- Optionally prepends a Telecommand header for use with the CCSDS Telecommand Encoder Module

Required Hardware

- None

Add Module Guidelines

System limitations determine the number of CLCW PROCESSOR MODULES that can be added to a desktop. The total maximum allowable number of modules in a desktop is 32 and this number can not be exceeded. Since this is a software module, the CPU and data rates also determine how many modules can simultaneously process data.

Required Input Source, Connections, and Events

INPUT SOURCE: The CLCW Processor Module accepts buffers posted from other modules. Ordinarily this buffer would contain a Command Link Control Word (CLCW) encoded in a telemetry stream from a spacecraft.

CONNECTIONS: The CLCWs are posted to Port 1.

EVENTS: No events required. No events generated.

Module Functional Description

The Command Link Control Word (CLCW) is part of the CCSDS COP-1 standard for telecommand uplink. It is the status feedback from the spacecraft. The CLCW is four bytes.

The CLCW Processor Module extracts the CLCW from a user-defined offset in the telemetry stream. It posts the CLCW to Port 1. To reduce unnecessary traffic, the module can be configured to post only the changes in CLCWs.

The CLCW Processor Module works with the CCSDS Telecommand Module (COP-1). In order to post the CLCWs to the CCDSD Telecommand Module, the CLCW Processor Module must prepend the Telecommand Header to the data. A required field in the telecommand header is the Spacecraft Identifier (SCID). Since the CLCW does not contain the SCID, the CLCW Processor Module may be configured use a set SCID or to automatically read the SCID from the header of the buffer posting the telemetry data.

Input Data Format

The input data buffers require telemetry data containing a CLCW at a fixed offset.

Output Data Format

The output data (Port 1) contains the 4-byte CLCW with an optional telecommand header for use with the Telecommand Encoder Module. The fields of the 4-byte CLCW are shown below:

Name	Bits	Description
Contol Word Type "0"	1	Always zero for CLCW
CLCW Version "00"	2	00 = Version 1
Status Field	3	Mission specific
COP in Effect	2	01 = COP-1
VCID	6	Virtual Channel Identifier associated with this CLCW
Spare	2	Reserved. Set to "00"
No RF Flag	1	0 = Physical channel is ready, 1 = Not ready
No Bit Lock Flag (opt.)	1	0 = Bit lock achieved, 1 = No bit lock
Lockout Flag	1	1 = Type A packets locked out, 0 = Not locked out
Wait Flag	1	1 = Cannot accept A packets, 0 = Cleat to send
Retransmit Flag	1	1 = Retransmit A packet, 0 = No retransmit
FARM B Counter	2	2 lsb's of the FARM B Counter
Spare	1	Reserved. Set to "0"
Report Value	8	Value of next expected A packet sequence number

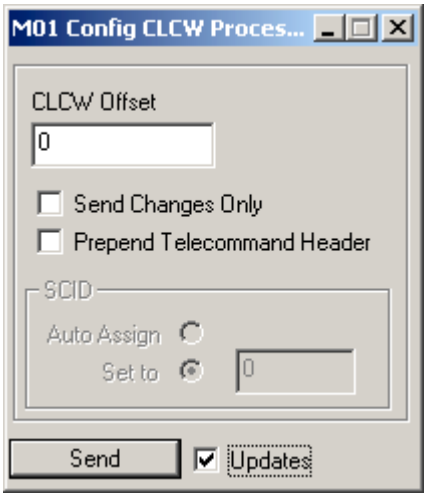
Module Window

When CLCWProcessor.dll is added to a desktop, the CLCW Processor dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 24-1: CLCW PROCESSOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CLCW PROCESSOR MODULE Board Parameters sub-menu.



Module 24-2: Config CLCW PROCESSOR MODULE Window

Module 24-3: CLCW PROCESSOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
CLCW Offset	Integer	32 bits	The fixed offset to the CLCW.
Send Changes Only	Check Box	Boolean	When checked, only post changes to incoming CLCWs to the output port
Prepend Telecommand Header	Check Box	Boolean	When checked, prepend a telecommand header to the CLCW output data for use with the Telecommand Encoder Module.
SCID			
Auto Assign	Radio Button	Boolean	Assign the SCID to the prepended telecommand header by reading the information from the header of the buffer posted the module's input port.
Set to	Radio Button	Boolean	Assign the given SCID to the prepended telecommand header.
	Identifier	6 bits	

Extended Options Window Definition

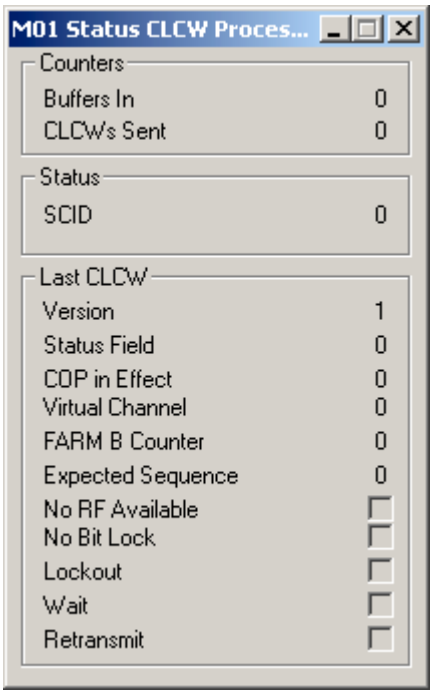
There is no extended options window for the CLCW PROCESSOR MODULE.

Configuration Notes

The Telecommand Encoder Module is included in the standard module set. Frequently, this module is intended for use with the CCSDS Telecommand Module. The CCSDS Telecommand Module software is not part of the standard Avtec PTP for Windows install and must be purchased separately (note that the CCSDS Telecommand Module window DOES appear in the module selection list of a standard Avtec PTP for Windows install, but the software is not present to support it).

Status Window Definition

When you click the Status button on the CLCW PROCESSOR MODULE window, a Status CLCW PROCESSOR window appears, as shown below.



Module 24-4: Status CLCW PROCESSOR MODULE Window

Module 24-5: CLCW PROCESSOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
COUNTERS			
Buffers In	Counter	32 bits	Count of number of buffers posted to the module
CLCWs Sent	Counter	32 bits	Count of number of CLCWs actually posted to Port 1
Status			
SCID	Identifier	6 bits	Spacecraft Identifier used if the telecommand headers are prepended
Last CLCW			
Version	Version	1-4	Version of CLCW
Status Field	None	3 bits	Mission-specific status bits
COP in Effect	Buffers	2 bits	COP version
Virtual Channel	Identifier	6 bits	VCID associated with CLCW
FARM B Counter	Counter	2 bits	2 lsb's of FARM B Counter

Parameter	Unit	Range or Values	Description
Expected Sequence	Sequence Num	8 bits	Next expected A Packet sequence number for this VC
No RF Available	Check Box	Boolean	Value of last No RF Available bit
No Bit Lock	Check Box	Boolean	Value of last No Bit Lock bit
Lockout	Check Box	Boolean	Value of last Lockout Bit
Wait	Check Box	Boolean	Value of last Wait bit
Retransmit	Check Box	Boolean	Value of last Retransmit bit

Module 25 CMOD MODULE

MODULE NAME: Cmod.dll

Key Functions

- Allows control of the ATSC built Command Modulator board using ATSC supplied software.
- Reads status information from the ATSC built Command Modulator board using ATSC supplied software.

Required Hardware

- The following hardware must be installed in the Avtec PTP for Windows system:
  - ✓ CMOD card, one per system

Add Module Guidelines

One card per computer and one software module instance.

Required Input Source, Connections, and Events

INPUT SOURCE: Command data is received from another module.

CONNECTIONS: Another module must feed command data to this module.

EVENTS: No events are expected or generated.

Module Functional Description

This module calls functions supplied by ATSC to setup and control the CMOD board. Currently, only one board may be inserted into a system because that software is hard-coded to use interrupt 11 and I/O address 0x300. Command data to be modulated and uplinked is sent via the ISA bus to the card. Status is reported back at the completion of each called function. The status reported from the calls is available on the GUI or in the Remote Interface Library (RIL).

When the module is enabled, the card is instructed to reset and initialize. An operating system is loaded from a user-specified file to the card. The FPGA is also loaded from a user specified file to the card. Finally, a block of data containing the user chosen setup parameters is downloaded to the board. The board is then commanded to start. If idle modulation is enabled, it begins at this time. When a buffer of command data is received, it is downloaded to the board. It has been noticed that the board modulates one extra bit at the beginning. The state of this bit cannot be determined.

Input Data Format

The data length, and data content of the buffer is passed to the board. The software prepends a two byte binary length code to the data, which is not uplinked.

Output Data Format

No data is output from this module.

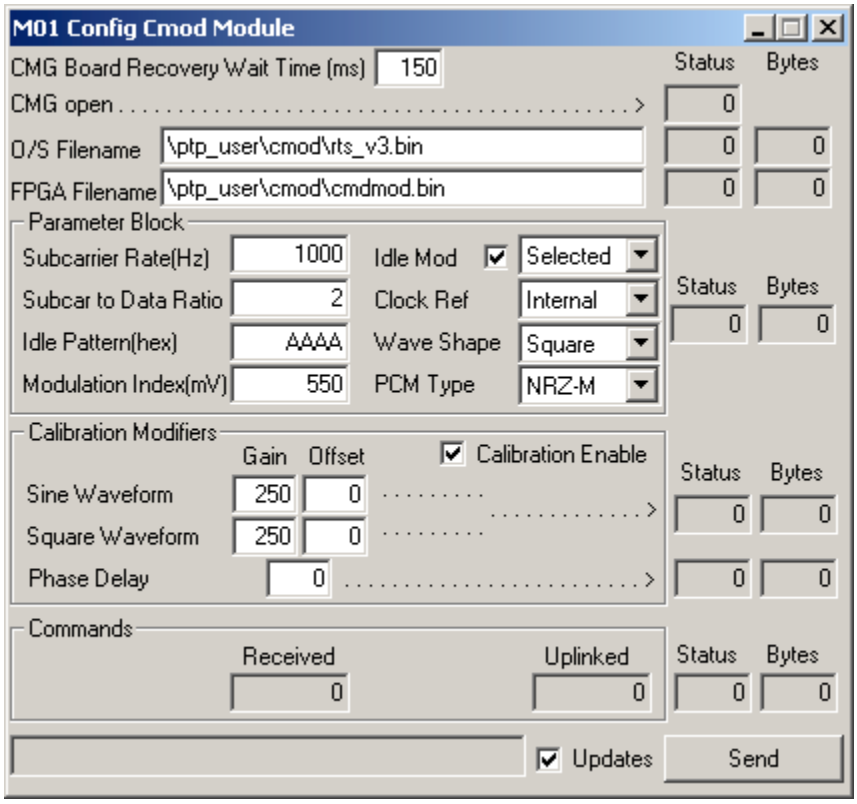
Module Window

When Cmod.dll is added to a desktop, the CMOD dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 25-1: CMOD MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CMOD MODULE Board Parameters sub-menu.



Module 25-2: Config CMOD MODULE Window

Module 25-3: CMOD MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
CMG Board Recovery Wait Time	ms	10 to 2000	Time period to wait between board control function calls. This period allows the card to finish the function before a new control instruction is sent.
CMG Open		0 or not 0	Status from ATSC driver open call; 0 is a good open
O/S Filename			Filename of the operating system that should be loaded onto the board; followed by the status and byte count returned from the ATSC driver call
FPGA Filename			Filename of the FPGA code that should be loaded onto the board; followed by the status and byte count returned from the ATSC driver call
PARAMETER BLOCK			
Subcarrier Rate (Hz)	Hz		Subcarrier rate in hertz
Subcar to Data Ratio			Data to subcarrier ratio
Idle Pattern	Hex		8 bit idle pattern; repeated
Modulation Index	mV	0 to 550	550 = 5.5 volts

Parameter	Unit	Range or Values	Description
Idle Mod	Selectable Setup State	No Idle Selected	Turns on or off idle modulation; the checkbox set/clears bit 8 in the parameter block
Clock Ref	Selectable Setup State	Internal External	Selects internal or external clock
Wave Shape	Selectable Setup State	Square Sine	Selects wave shape
PCM Type	Selectable Setup State	NRZ-M, NRZ-S, NRZ-L BIO-M, BIO-S, BIO-L	PCM Modulation type
CALIBRATION MODIFIERS			
Calibration Enable			Causes the board to accept calibration parameters
Sine Waveform			Gain and Offset parameters for the sine waveform calibration; followed by the status and byte count from the ATSC driver call
Square Waveform			Gain and Offset parameters for the square waveform calibration; followed by the status and byte count from the ATSC driver call
Phase Delay			Phase calibration; followed by the status and byte count from the ATSC driver call
COMMANDS			
Received			Number of commands received by the software module
Uplinked			Number of command successfully uplinked as reported by the CMOD card and driver

Extended Options Window Definition

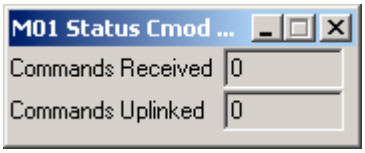
There is no extended options window for the CMOD MODULE.

Configuration Notes

There are no configuration notes for the CMOD MODULE.

Status Window Definition

When you click the Status button on the CMOD MODULE window, a Status CMOD window appears, as shown below.



Module 25-4: Status CMOD MODULE Window

Module 25-5: CMOD MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Commands Received	Count	Increments by one for every command received by module	Number of commands received by the software module
Commands Uplinked	Count	Increments by one for every command output from module	Number of command successfully uplinked as reported by the CMOD card and driver

Module 26 COM PORT TRANSCEIVER MODULE

MODULE NAME: Comport.dll

Key Functions

- Sets up and allows data transfer to and from asynchronous serial port
- Reports number of bytes transferred to and from the port

Required Hardware

- No special hardware required.

Add Module Guidelines

ComPorts cannot be shared between modules. Therefore, one module can be set up for each ComPort available to use. Do not exceed 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from any other module or from the serial port (bi-directional module).

CONNECTIONS: Output to port one or to serial port. (bi-directional module).

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The Com Port Transceiver Module is used to send and receive buffers of data via the PC's Com Ports. It is a bi-directional module.

Module sends data out an asynchronous port.

Input Data Format

Any data

Output Data Format

Any data

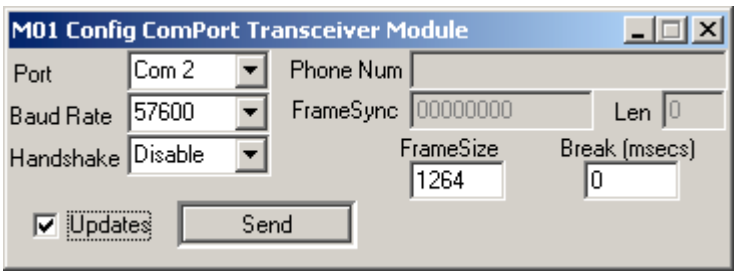
Module Window

When Comport.dll is added to a desktop, the Com Port Transceiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 26-1: COM PORT TRANSCEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure COM PORT TRANSCEIVER MODULE Board Parameters sub-menu.



Module 26-2: Config COM PORT TRANSCEIVER MODULE Window

Module 26-3: COM PORT TRANSCEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Port	Selectable Setup	Com 1 Com 2 Com 3 Com 4	Select Com Port 1, 2, 3, or 4 Ports as NT has assigned. GUI supports four comports, but RIL allows support of more.
Phone Num			Future
Baud Rate	Selectable Setup	110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600	Set data rate for Com Port (110 to 57,600 bps) Configure to match the serial device to which it is connected
Frame Sync	N/A	N/A	Unused
Len	N/A	N/A	Unused
Handshake	Selectable Setup	Enable Disable	Enable or Disable Handshaking for sending and receiving data DTR-(data terminal ready) is enabled/disabled. If enabled, do not transmit until DRT is asserted.
Frame Size	bytes	16 Kbyte maximum	Applies to the receive-side only. Specify Frame Size for incoming or outgoing data

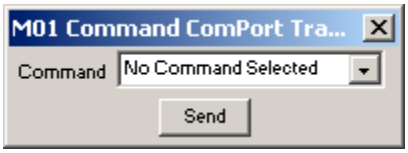
Parameter	Unit	Range or Values	Description
Break (msecs)	msec	0 to 1 second	Inserts a break signal. A non-zero value inserts a non-transition period of break time and issues a hardware-level signal called break (framing error).

Extended Options Window Definition

There is no extended options window for the COM PORT TRANSCEIVER MODULE.

Configuration Notes

When you click the Command button on the COM PORT TRANSCEIVER MODULE window, a Command COM PORT TRANSCEIVER window appears, as shown below.



Module 26-4: Command COM PORT TRANSCEIVER MODULE Window

Module 26-5: COM PORT TRANSCEIVER MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	N/A	N/A	Indicates that commanding is in a disabled state and issuing a send causes no response
Dial	N/A	N/A	Currently unavailable

Status Window Definition

When you click the Status button on the COM PORT TRANSCEIVER MODULE window, a Status COM PORT TRANSCEIVER window appears, as shown below.

Module 26-6: Status COM PORT TRANSCEIVER MODULE Window

Module 26-7: COM PORT TRANSCEIVER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Tx Count	bytes	Incrementing count	It is the number of bytes output to serial port.

Parameter	Unit	Range or Values	Description
Tx Que Oflows	overflows	Incrementing count	Transmitter queue overflow. Increments when data is input to module faster than can be output to port. Size of queue is six buffers (whatever size buffer is input)
Rx Count	bytes	Incrementing count	Displays the total number of bytes received from port.

Module 27 COMMAND VERIFY MODULE

MODULE NAME: CommandVerify.dll

Key Functions

- Compares uplink data to radiated data to validate data radiated from the antenna feedhorn.

Required Hardware

- No hardware is required for this module, however, necessary hardware and modules are needed on the desktop to supply both the expected uplink and the actual radiated data to this module. Cable connections are necessary to provide the radiated data to a demodulator, bitsync, and serial input card. An associated Serial Input Module is needed on the desktop to send the data to this module.

Add Module Guidelines

One copy of this module is necessary per set of uplink and radiated stream to be validated.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives data from two other modules, the expected uplink stream and the actual radiated stream.

CONNECTIONS: The two input modules must be connected to this module. No output module connections are necessary.

EVENTS: No events are expected or generated.

Module Functional Description

The COMMAND VERIFY MODULE compares two command streams and outputs a log file of command statistics.

In typical use, this module is used to compare a ground station Avtec PTP for Windows generated command uplink to the actual commands that were radiated from the command uplink. The user specifies the module that is transmitting the commands, the module that is receiving the radiated commands, and a time period equivalent to the estimated delay between the two streams. The module then outputs a log file containing statistics on each frame transmitted and radiated.

The radiated command data is matched up with the expected uplink data. This match up is order dependent. A user specified timeout is imposed to halt the search for the radiated data. First, the length of the uplink data is compared to the length of the radiated data. If the

lengths are not equal, an error is generated and the comparison process is halted. Results are reported in the log file. If the lengths agree, a byte-wise comparison is used to validate the radiated data. If the data from the uplink stream exactly agrees with the data from the radiated stream, the radiated stream is considered validated. If the radiated data does not exactly agree with the data from the uplink stream, the bits in the incorrect byte are examined, one at a time to find the first incorrect bit. The error is reported in the log file.

The user is required to specify with module is supplying the uplink data and which is supplying the radiated data. The module numbers are check to insure that the user does not accidentally specify the same module number for both streams. Note the command verify module cannot tell the difference between the streams on its own, since the just look like buffers with data in them, by the time that they are received by this module. The user has to insure that the correct module number is specified for both the uplink data stream and the radiated stream. The user must also specify the timeout to wait for the radiated data. If no radiated data is received within the timeout window, an error is generated.

The user can specify whether to log all data, just errors, or not to log at all. Results are also reported on the status window and are available using the Remote Interface Library (RIL) to other processes.

Input Data Format

This module expects two inputs, one from the user specified uplink module, and the other from the user specified radiation (probe verification) modules. Any other data received is ignored.

Output Data Format

No data is output from this module. An ASCII log file is generated, if desired, containing errors. Optionally the user may elect to include data which validated successfully. The log is coded with special character sequence to aid the user. These special characters allow the user to search through the text file quickly to locate the desired information. Some of the characters continue to the end of the user specified line length to provide easy visual separation. The special characters are:

$ $	(dollar, space, dollar)	File Header, either at beginning of file, or the point of appended information. The "dollar, space, dollar, space, …" pattern continue to the end of the line, after the header information.
. .	(period, space, period)	File Setup Information, captures user selections from the config window.
+ +	(plus, space, plus)	Beginning of a successful verification sequence. It is followed by the status of both buffers and a dump (in the

		user specified base) of the buffer content. The "plus, space, plus, space, …" pattern continues to the end of the line.
# #	(pound, space, pound)	Beginning of a failed verification sequence. This can be either a length mismatch or a data content mismatch. All available information is listed according to user specifications. The "pound, space, pound, space, …" pattern continues to the end of the line.
* *	(star, space, star)	Missing buffer. Reported as part of a timeout. All available information is listed according to user specifications.
= =	(equal, space, equal)	End of sequence marker. Follows either a failed or successful verification sequence. Acts as a separator between sequences. The "equal, space, equal, space, …" continues to the end of the line.
@ @	(at, space at)	File Trailer, either at the end of file, or just before the point of appended information. The "at, space, at, space, …" continues to the end of the line.

A sample log is inserted below. Assume that it is part of a sample pass. Comments are larger font, underlined, bold and italic, and are not part of the file itself.

File Header:

```
$ $ \ptp_user\logs\cmdvfy.txt Enabled (Init, Hex, All) $ $ $ $ $ $ $ $ $ $ $ $
. . Byte Numbering [0,1,2,...]; Bits of each byte Numbered [0,1,2,...,6,7]; Bits of
buffer Numbered [0,1,2,...]
```

Good Verification:

```
+ + Good Sequence; delta_t = plus 6271933 uS  + + + + + + + + + + + + + + + + +

023:16:46:08.952.337 -- 5 bytes Module 2 [Transmitted]
01 02 03 04 05

023:16:46:15.224.270 -- 5 bytes Module 3 [Radiated]
01 02 03 04 05

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

Bad Verification; wrong length:

```
# # Length Mismatch # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #

023:16:46:31.468.169 -- 4 bytes Module 2 [Transmitted]
01 02 03 04

023:16:46:33.187.548 -- 5 bytes Module 3 [Radiated]
01 02 03 04 05

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

Bad Verification; Timeout on Radiation Data:

```
# # Missing Radiation Data; No comparison; t/o = 30000 ms # # # # # # # # # #

023:16:47:00.827.324 -- 5 bytes Module 2 [Transmitted]
```

```
01 02 03 04 05

* * No data received from Module 3 [Radiated]

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

Good Verification:

```
+ + Good Sequence; delta_t = plus 1179870 uS  + + + + + + + + + + + + + + + +

023:16:47:37.353.422 -- 5 bytes Module 2 [Transmitted]
01 02 03 04 05

023:16:47:38.533.292 -- 5 bytes Module 3 [Radiated]
01 02 03 04 05

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

Bad Verification; Data bit error

```
# # Data Error; bit 6 of byte 3 (abs bit 30) # # # # # # # # # # # # # # # # #

023:16:47:55.738.215 -- 5 bytes Module 2 [Transmitted]
01 02 03 04 05

023:16:47:56.764.153 -- 5 bytes Module 3 [Radiated]
01 02 03 06 05

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

Good Verification:

```
+ + Good Sequence; delta_t = plus 1022259 uS  + + + + + + + + + + + + + + + +

023:16:48:22.197.300 -- 5 bytes Module 2 [Transmitted]
01 02 03 04 05

023:16:48:23.219.559 -- 5 bytes Module 3 [Radiated]
01 02 03 04 05

= = Sequence End = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
```

File Trailer:

```
 @ @ \ptp_user\logs\cmdvfy.txt Closed @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @ @
```

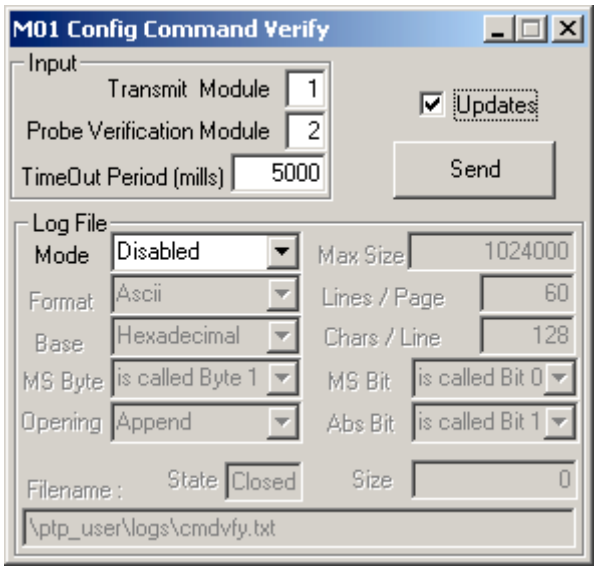
Module Window

When CommandVerify.dll is added to a desktop, the Command Verify dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 27-1: COMMAND VERIFY MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure COMMAND VERIFY MODULE Board Parameters sub-menu.



Module 27-2: Config COMMAND VERIFY MODULE Window

Module 27-3: COMMAND VERIFY MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			
Transmit Module	integer	1 to 32 (max # modules/desktop = 32)	Identifies input data stream by the source module number. This is the command stream that represents Avtec PTP for Windows uplinked commands.
Probe Verification Module	integer	1 to 32	Identifies input data stream by the source module number. This is the command stream that represents radiated commands. The radiated data source is usually a tap at the antenna feedhorn.
Time Out Period	Milli-seconds	0 to 30 seconds, specified in milliseconds	Defines the maximum time delay between Transmit Module command data stream and the receipt of the Probe Verification Module echoed data stream. If the Probe Verification module data arrives late, an error would be generated at the timeout stating that the radiated data was missing; and a second error would be generated stating that radiated data was received with no matching uplink data stream. These two errors would be reported on the status window and in the users' log file. It is up to the user to analyze the log file to determine that the radiated data was either late or the timeout value was specified too small.
LOG FILE			

Parameter	Unit	Range or Values	Description
Mode	Selectable Setup	Disabled Log Errors Only Log All Buffers	In the Disabled mode, no log file is generated. In the Log All Buffers mode, all uplink and radiated data is recorded with its verification status. Log Errors Only mode, just logs the errors with the associated data to the log.
Format	Selectable Setup	ASCII	Defines the file format. Currently, only ASCII is the only format available.
Base	Selectable Setup	Hexadecimal Binary Octal Decimal	Specifies the number base of the output buffers. The base applies to all data logged in the file.
MS Byte	Selectable Setup	Is called Byte 1 Is called Byte 0	Specifies how the user numbers the bytes in the uplink and radiated data. "Is called Byte 0" starts numbering the most significant (first transmitted) byte in the buffer as number zero. "Is called byte 1" starts numbering the bytes starting with number one. This input is used to specify the first byte wrong number [with a bit number] of the radiated buffer (in the log file) when a difference between the uplink data and the radiated data is found.
Opening	Selectable Setup	Append Init	Defines how the file is written. If set to Append, the file is opened and data is appended to existing file. If set to Init, the file is opened, and, if previous data exists, it is overwritten (lost).
Filename	Text box entry		Defines the file to be written by filename, directory path, and logical disk. The user must take care to assure that the path actually exists. An error is generated if the specified path and file cannot be written.
State	Display only	Open /Closed	Displays the state of the log file.
Size	Display Only	Decimal Numeric	Displays the actual size in bytes of the current log file.
Max Size	Decimal	1024 to 2000000000 (1kb to 2gb)	Specifies the maximum size of the log file. Once the size is reached, the file is automatically closed, and logging is disabled. Note that the recording stops early to allow a trailer to be written to the file.
Lines/Page	Decimal	40 to 256	Specifies the size, in lines, of a printed page. A formfeed character, 0x0c, is written to the file each time this line count is reached. If the user intends to print out the log information, this should determine this based on the size of the paper and the printer type in use.
Chars/Line	Decimal	80 to 256	Specifies the line length, in characters, of each line in the log file. If the user intends to print out the log information, this should determine this based on the size of the paper and the printer type in use.

Parameter	Unit	Range or Values	Description
MS Bit		Is called Bit 0 Is called Bit 1 Is called Bit 7 Is called Bit 8	Specifies the bit numbering of each byte. Some users number the bits differently. The MS (most significant) bit is the bit first transmitted, regardless of the numbering convention. "Is called Bit 0" states that the bits are numbered left to right, as 0-1-2-3-4-5-6-7 within a byte, most significant (first transmitted) bit first. "Is called Bit 1" states that the bits are numbered left to right, as 1-2-3-4-5-6-7-8 within a byte, most significant (first transmitted) bit first. "Is called Bit 7" states that the bits are numbered right to left, as 7-6-5-4-3-2-1-0 within a byte, most significant (first transmitted) bit first. "Is called Bit 8" states that the bits are numbered right to left, as 8-7-6-5-4-3-2-1 within a byte, most significant (first transmitted) bit first. This input is used to specify the first bit wrong number [within a byte] of the radiated buffer (in the log file) when a difference between the uplink data and the radiated data is found.
Abs Bit		Is called Bit 1 Is called Bit 0	Specifies the absolute bit numbering convention for sequential bits in the buffer without regard to byte boundaries. [For example, if the buffer is 100 bytes long, the bits could be numbered '0 to 799' or '1 to 800'.] "Is called Bit 1" specifies that the most significant bit (first transmitted) is numbered bit 1. "Is called Bit 0" specifies that the most significant bit (first transmitted) is numbered bit 0. The absolute numbering continues sequentially thru the last bit in the buffer. This input is used to specify the first bit wrong number of the radiated buffer (in the log file) when a difference between the uplink data and the radiated data is found.

Extended Options Window Definition

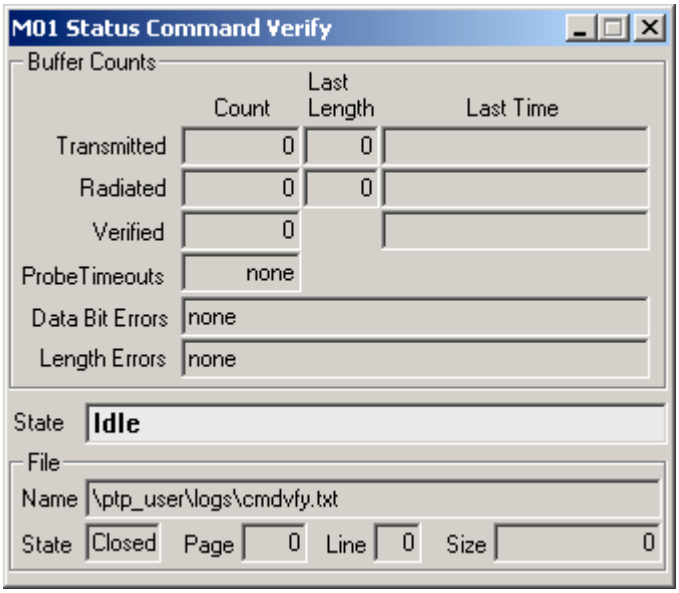
There is no extended options window for the COMMAND VERIFY MODULE.

Configuration Notes

There are no configuration notes for the COMMAND VERIFY MODULE.

Status Window Definition

When you click the Status button on the COMMAND VERIFY MODULE window, a COMMAND VERIFY window appears, as shown below.



Module 27-4: Status COMMAND VERIFY MODULE Window

Module 27-5: COMMAND VERIFY MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
BUFFER COUNTS			
Trans-mitted		Decimal or Time	1) Count: Number of buffers received from the Transmit module. 2) Last Length: Length, in bytes, of the last received transmit buffer. 3) Last Time: Buffer time of the last received transmit buffer.
Radiated			1) Count: Number of buffers received from the Probe Verification (radiated) module. 2) Last Length: Length, in bytes, of the last received Probe Verification buffer. 3) Last Time: Buffer time of the last received Probe Verification buffer.
Verified			1) Count: Number of successful comparisons between a transmit buffer and a Probe Verification buffer. 2) Last Time: Buffer Time of the last successful comparison. The radiated time is displayed.
Probe Timeouts			Count of timeouts that have occurred. No Probe Verification buffer was received by the Command Verify module prior to the user specified timeout (milliseconds).
Data Bit Errors			Count of comparisons which resulted in a data mismatch between the transmitted data and the Probe Verification data. An error can only be reported if the buffers are the same length. No comparison is made if the buffers are of different lengths.

Parameter	Unit	Range or Values	Description
Length Errors			Count of Length errors which occurred between the transmitted data and the Probe Verification data. Note that if the buffers are not the same length, no further error checking is done, so that there would not also be a "Data Bit Error"
State	Discrete State Text	Idle Awaiting Transmit Awaiting Radiation Comparing Buffers	"Idle" is reported if the module is not enabled. No processing of any input occurs. "Awaiting Transmit" is reported when the module is awaiting an uplink buffer. This is the normal state, and shows that the module is ready to receive a transmit buffer and its' radiation buffer to compare. A queue is maintained in case a second transmit buffer is received prior to the Probe Verification data for the first transmit buffer. "Awaiting Radiation" is reported when the module has received uplink data and is waiting for the radiation buffer which should be compared to the already received transmit buffer. If no buffer is received prior to the user specified timeout, the verify process generates an error; and the module proceeds to the "Awaiting Transmit" state again; assuming that the Probe Verification buffer will not be received. If the Probe Verification buffer is received after the timeout, a second error is reported, because the is no transmit buffer pending verification. "Comparing Buffers" is reported when the module is comparing a transmit buffer to a Probe Verification buffer. Note that the console gui is updated once per second. Not all states that the verify process executes can be displayed. Only the state at the time the window is refreshed will be displayed.
FILE			
Name	filename		Lists the name of the file and its path that is in use as the log file.
State	Discrete State Text		Status of file: open or closed. This state is also reported in the Config window.
Page	decimal		Lists which page is currently being written to in the log file.
Line	decimal		Lists which line number of a specific page which was last written in the log file.
Size	decimal		Lists the current size, in bytes, of the file.

Module 28 CONVOLUTIONAL ENCODER MODULE

MODULE NAME: ConvolutionalEncoder.dll

Key Functions

- Convolutional encodes any input stream
- Programmable polynomials.
- Programmable tie backs.
- Outputs data encoded according to user-defined setup

Required Hardware

- This module requires no special hardware be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of CONVOLUTIONAL ENCODER MODULES may be loaded in a desktop. Limitations are system imposed. The maximum number of modules allowed in a desktop is 32 and CPU usage should be monitored since this is a software module.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide the input data.

CONNECTIONS: Encoded data is output on port 1.

EVENTS: Module may be run without event connections in burst/command mode. In continuous mode, events must be set up to propagate from the output module to the CONVOLUTIONAL ENCODER MODULE, and then from the CONVOLUTIONAL ENCODER MODULE to the module that is providing the data on the encoder input port. With the propagated event setup, the Convolutional Encoder reduces the requested frame size by the EncodeRate.

Module Functional Description

This module performs Rate ½ and Rate 1/3 convolutional encoding on an input data stream, and then outputs the encoded data.

Convolutional encoding parameters are user-configurable.

Input Data Format

There is no specific input data format.

Output Data Format

Input data size is expanded by 2x or 3x.

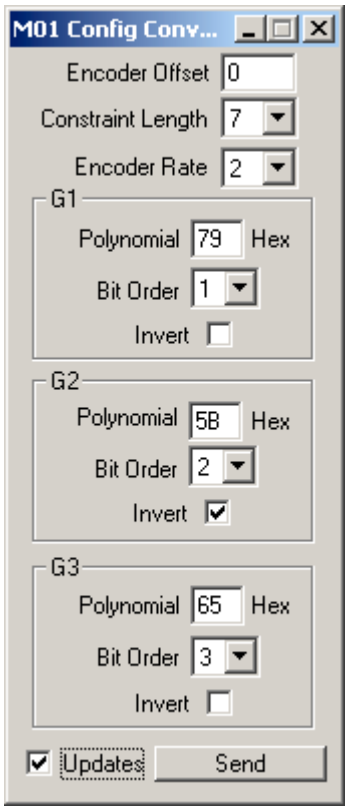
Module Window

When ConvolutionalEncoder.dll is added to a desktop, the Convolutional Encoder dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 28-1: CONVOLUTIONAL ENCODER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CONVOLUTIONAL ENCODER MODULE Board Parameters sub-menu.



Module 28-2: Config CONVOLUTIONAL ENCODER MODULE Window

Module 28-3: CONVOLUTIONAL ENCODER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Encoder Offset	**Bytes**	0 to 65535	Allows definition of bytes that are not encoded at the beginning of each data unit that is input. It leaves Offset bytes un-encoded at the beginning of the received buffer.
Constraint Length	**Units**	7,6,5,4,3,2	Defines constraint length.
Encoder Rate	**Units**	2 or 3	This version supports rate ½ and rate 1/3 encoding.
G1			
Polynomial	Hex	0 to 0xFF	
Bit Order	Selectable Setup State	1,2,3	
Invert	Selectable Setup State	Enable Disable	Enables symbol inversion for G1. Setup is a checkbox: insert a check to enable and remove the check to disable.
G2			
Polynomial	Hex	0 to 0xFF	
Bit Order	Selectable Setup State	1,2,3	
Invert	Selectable Setup State	Enable Disable	Enables symbol inversion for G2. Setup is a checkbox: insert a check to enable and remove the check to disable.
G3			
Polynomial	Hex	0 to 0xFF	
Bit Order	Selectable Setup State	1,2,3	
Invert	Selectable Setup State	Enable Disable	Enables symbol inversion for G3. Setup is a checkbox: insert a check to enable and remove the check to disable.

Extended Options Window Definition

There is no extended options window for the CONVOLUTIONAL ENCODER MODULE.

Configuration Notes

There are no configuration notes for the CONVOLUTIONAL ENCODER MODULE.

Status Window Definition

There is no status window for the CONVOLUTIONAL ENCODER MODULE.

Module 29 CPU TIMER MODULE

MODULE NAME: CPUTimer.dll

Key Functions

- Generates end-of-frame interrupts to meter the output of data
- Does not require hardware to provide end-of-frame interrupts
- Optionally throughputs data buffers: output is at the configured rate, but no data buffering is provided
- Provides programmable event (frame) length and rate
- Provides event to any other module

Required Hardware

- This module requires not special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of CPU TIMER MODULES may be included in the user's desktop (32 total modules per desktop is permissible).

Required Input Source, Connections, and Events

INPUT SOURCE: When used to generate events, this module requires no data input. When used to meter data, it accepts any buffer data input from another module.

CONNECTIONS: No data connections are needed if it is used as an event source only. If it is being used to meter data, it outputs to port one, and the output must be connected to another module's input.

EVENTS: This module generates events for other modules. To use this function, an event connection to one or more modules is required.

Module Functional Description

The CPU TIMER MODULE generates events for other modules based on a user-specified interval. The CPU TIMER MODULE has an event output but no data output.

The CPU TIMER MODULE can be used to simulate end-of-frame interrupts when no serial I/O hardware is available. For example, the CPU Timer event could be connected to the Virtual Channel Simulator to generate frames at a particular rate for transmission over the network.

As a secondary function, the CPU TIMER MODULE can be used to rate-meter data, but it does not provide any buffer capability, so it must be used carefully. If data is input to this module,

it is output on port one at whatever rate for which it is configured regardless of input rate. It can only accept data input at whatever rate it is set to output, so it either causes the input source to loose data or meter its data if it has that capability. Besides being used for diagnostic purposes, the most common use of this function is to meter network input. When set to receive input from a network module, the CPU timer accepts data at its configured rate. Since this causes the network module to only send data at that rate, the TCP buffers become full, and then TCP stops sending data acknowledgements, which causes the network source to meter its data.

Input Data Format

This module requires no input when used as an event source. It can accept any Avtec PTP for Windows buffer input.

Output Data Format

There is no data output when this module is used as an event source. If buffer data is input, it outputs each buffer unchanged.

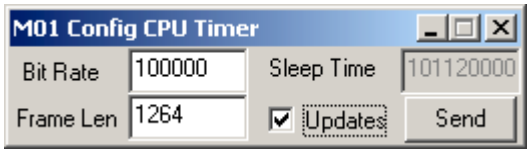
Module Window

When *CPUTimer.dll* is added to a desktop, the CPU Timer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 29-1: CPU TIMER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure CPU TIMER MODULE Board Parameters sub-menu.



Module 29-2: Config CPU TIMER MODULE Window

Module 29-3: CPU TIMER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Bit Rate	bps	System dependent	Specifies the simulated bit rate in bps. For event generation, determines the event generation rate. If data is input to the module, this defines the output rate on port one.
Frame Len	bytes		Specifies the simulated frame length when used for event generation (EOF interrupt). Must match the input buffer size when used for data output.
Sleep Time	nsecs	64-bit value	Displays the period between events in nanoseconds

Extended Options Window Definition

There is no extended options window for the CPU TIMER MODULE.

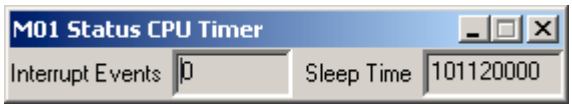
Configuration Notes

The Bit Rate and Frame Len parameters are used to calculate the number of milliseconds between events. This module could be connected to a CCSDS Virtual Channel Simulator or a Time Division Multiplex (TDM) Simulator to cause frames to be output from those modules at a given rate. The expected frame length specified in the CPU TIMER MODULE carries forth to the module receiving the events just as the frame length from a serial input module. This module effectively simulates the presence of serial hardware end of frame interrupts.

The performance of the CPU timer is dependent on the availability of other system resources. Although the time between interrupts in maintained in nanoseconds, and a best effort is made to generate events at the expected rate, higher priority tasks and system driver can preempt CPU Timer from running and the generation of events can be compromised depending on system load, causing some jitter in the output data.

Status Window Definition

When you click the Status button on the CPU TIMER MODULE window, a Status CPU TIMER window appears, as shown below.



Module 29-4: Status CPU TIMER MODULE Window

Module 29-5: CPU TIMER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Interrupt Events	counter	Increments by one for each event	Displays count of events generated
Sleep Time	nsec	32-bit value	Displays time between events in nanoseconds

Module 30 DCS DEMOD MODULE

MODULE NAME: DCSDemod_BS.dll

Key Functions

- Controls the Data Capture Systems (DCS) Demodulator, BitSync and Viterbi unit.
- Displays the status of the unit.

Required Hardware

- DCS Demod/BitSync with Viterbi decoding unit.

Add Module Guidelines

Only one module is necessary to control the DCS Demod unit.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives text-based information from the DCS unit. The content of the text is displayed for the user and is available from the Remote Interface Library (RIL). The actual electrical interface is a National Instruments GPIB (General Purpose Interface Bus) NI-488.2. The card used is model AT-GPIB/TNT. It is an ISA-bus based card. An interrupt, I/O port address, and DMA channel all need to be set using jumpers on the card. The default card settings are: IRQ 11, I/O 2C0, and DMA 5. The DMA channel can be disabled if desired; it is not used by this module. Refer to the National Instruments documentation which was supplied with your GPIB card. This module was coded using the October 1996 edition of "Getting Started with Your AT-GPIB/TNT and the NI-488.2M Software for Windows-NT", part number 320648C-01. The GPIB function calls for this module were defined using the December 1996 edition of "NI-488.2M Function Reference Manual for Win32", part number 321038C-01.

CONNECTIONS: No module connections should be made with this module. A GPIB cable is required to establish connectivity between the DCS unit and the GPIB card installed in the computer. See the National Instruments documentation (enumerated above) for cable length restrictions, and correct procedures for establishing a GPIB network.

EVENTS: No events are expected or generated.

Module Functional Description

This module sends GPIB formatted ASCII text based commands to the DCS hardware unit

Input Data Format

Valid GPIB textual control and status text.

Output Data Format

Valid GPIB textual control and status text.

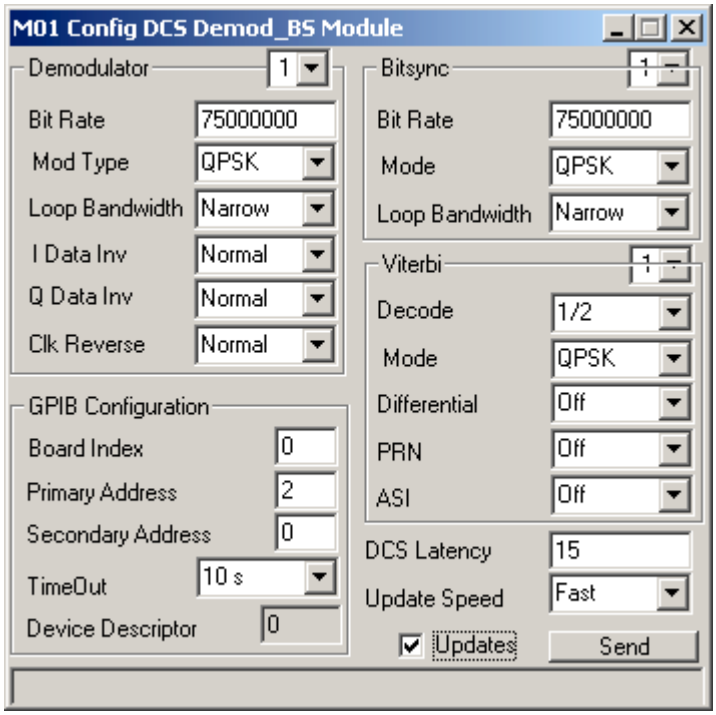
Module Window

When DCS Demod.dll is added to a desktop, the DCS Demod dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 30-1: DCS DEMOD MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DCS DEMOD MODULE Board Parameters sub-menu.



Module 30-2: Config DCS DEMOD MODULE Window

Module 30-3: DCS DEMOD MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
DEMODULATOR			
Demodulator	User selection	1	Selects the desired demodulator. As of this writing, there is only one demod to control.
Bit Rate	Decimal Hz	10 to 52000000	Selects demod expected bit rate. The demod will attempt to establish lock to this center frequency.
Mod Type	User selection	QPSK OQPSK BPSK UIQPSK(I>Q) UQQPSK(Q>I) EXTIN	Expected modulation type
Loop Bandwidth	User selection	Narrow Medium Wide	The demod will track data in both directions up to the amount specified. See the DCS documentation to determine the exact size definition of Narrow, Medium, or Wide.
I Data Inv	User Selection	Normal Invert	If set to invert, the data will be inverted, with respect to it's current state. Note that the demod can lock up in either polarity unpredictably when a dropout occurs.
Q Data Inv	User selection	Normal Invert	If set to invert, the data will be inverted, with respect to it's current state. Note that the demod can lock up in either polarity unpredictably when a dropout occurs.
Clk Reverse	User selection	Normal IQSwap	If set to IQSwap, the I and Q channels are exchanged.
GPIB CONFIGURATION			
Board Index	Decimal	Unrestricted	GPIB card number in computer. The National Instruments software driver allows the user to specify which board number is at a specific I/O port. Typically 0 if only one GPIB card is installed in the target computer.
Primary Address	Decimal	Unrestricted	GPIB address chosen by DIP switch on the back panel of the DCS hardware.
Secondary Address	Decimal	0	GPIB secondary address when required by hardware. Not currently used, but must be passed to the GPIB software calls.
Time Out	User Selection	None, 10us, 30us, 100us, 300us, 1ms, 3ms, 10ms, 30ms, 100ms, 300ms, 1s, 3s, 10s, 30s, 100s, 300s, 1000s	GPIB command response timeout. Only one command may be active at a time. This selection specifies the length of time to wait before giving up on a command, and allow the user to send another GPIB command. Commands are generated automatically when the Send button is pressed. Commands are also automatically generated to obtain status of the DCS unit.

Parameter	Unit	Range or Values	Description
Device Descriptor	Decimal	Display only	Shows the GPIB device descriptor assigned automatically by the GPIB driver. When a GPIB device is successfully opened, the National Instruments driver software returns a device descriptor that is used to reference that specific hardware for all reads and writes until a close command is preformed; when the descriptor is freed. An unsuccessful open is determined by the receipt of a descriptor value of 65535 or 4294967295, or minus one decimal.
BIT SYNC			
Bit Sync	User selection	1	Selects the desired bitsync to control. As of this writing there is only one bitsync in a DCS box.
Bit Rate	Decimal Hz	10 to 30000000	Defines the expected bit rate. A carrier is generated at this rate, and every attempt is made to phase lock the demodulated data to the carrier.
Mode	User selection	QPSK BPSK	Selects the expected phase shift key mode: Quadrature phase or Biphase.
Loop Bandwidth	User selection	Narrow, Medium, Wide	Selects the permissible loop bandwidth. This setting controls how far the carrier is allowed to deviate up or down from the user specified bit rate, attempting to maintain lock, and contiguous data output.
VITERBI			
Viterbi	User selection	1	Selects the Viterbi decoding unit number. As of this writing, only one decoder exists.
Decode	User selection	1/2	Selects the decode rate. As of the writing only 1/2 is available.
Mode	User selection	QPSK BPSK	Selects the expected phase shift key mode: Quadrature phase or Biphase.
Differential	User selection	Off On	If selected, enables differential decoding.
PRN	User selection	Off On	If selected, enables PRN (V.35 descrambler).
ASI	User selection	Off On	If selected, enables Alternate Symbol Inversion.
OTHER SETUP			
DCS Latency	Decimal	10 to 2000	Milliseconds to wait after a GPIB read is complete. This allows the hardware to turn around and expect a write next. This value should be set by the user based on the thruput of his network, and data loading on the Avtec PTP for Windows and the DCS box. Initially the default rate of 15 has been used successfully.
Update Speed	User selection	Fast, Normal, Slow	Console gui update rate. Fast updates the screen once in about 8 seconds. This is fastest acceptable update rate, and was determined experimentally. Normal causes an update once every 16 seconds. Normal should be used for high data rates, since the microprocessor in the DCS handles both data movement and gpib status updates. Slow provides an update once in 32 seconds.

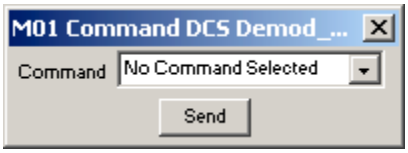
Parameter	Unit	Range or Values	Description
Blank line at bottom of gui	Display	Display	Error display. An errors received from the GPIB interface are translated into text and displayed here, including the I/O status; this error can be looked up in the NI documentation for a more detailed explanation of the problem. The module will continue to establish reliable communications until the module is disabled.

Extended Options Window Definition

There is no extended options window for the DCS DEMOD MODULE.

Configuration Notes

When you click the Command button on the DCS DEMOD MODULE window, a Command DCS DEMOD window appears, as shown below.



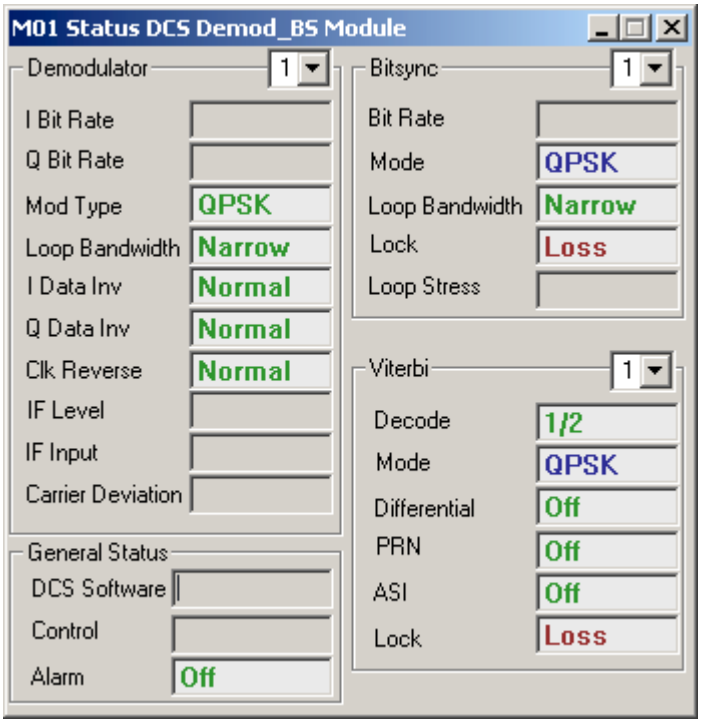
Module 30-4: Command DCS DEMOD MODULE Window

Module 30-5: DCS DEMOD MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected			Disables DCS immediate command output. Effectively no-ops the window.
Go remote			Command the DCS box to go to remote control. This commands is automatically generated by the mode at initialization. If the DCS box is set to local control, only status monitoring is available.
Go local			Commands the DCS box to go to local (front panel) control. No further commands will be accepted from the software. Status monitoring will continue, however.
Alarm Clear			Commands the DCS box to clear the alarm indicator on the front panel. This alarm illuminates to inform the local operator that the DCS power has been interrupted.
Demod 1 Step			Commands the DCS box to change the demod parameters. One parameter at a time is step-ed through its selections. This commands does not seem to work remotely.

Status Window Definition

When you click the Status button on the DCS DEMOD MODULE window, a Status DCS DEMOD window appears, as shown below.



Module 30-6: Status DCS DEMOD MODULE Window

Module 30-7: DCS DEMOD MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
DEMODULATOR			
Demodulator	User selection	1	Selects the desired demodulator for status readout. As of this writing, there is only one demodulator present.
I Bit Rate	Hz		Demod bit rate set use for the I channel. This is the bit rate setting, and is not representative of any data reception.
Q Bit Rate	Hz		Demod bit rate set use for the Q channel. This is the bit rate setting, and is not representative of any data reception.
Mod Type		QPSK OQPSK BPSK UIQPSK(I>Q) UQQPSK(Q>I) EXTIN	Modulation type. This is the demod setting, and is not representative of any data reception.
Loop Bandwidth		Low, Medium, High	Loop Bandwidth setting. This defines how closely the VC0 will track to the specified input rate. Low allows faster acquisition; High allows for more deviation (stress) of the bitrate.
I Data Inv		Normal, Invert	I data inversion setting. Not indicative of any data received, since the demod can lock to either polarity.
Q Data Inv		Normal, Invert	Q data inversion setting. Not indicative of any data received, since the demod can lock to either polarity.

Parameter	Unit	Range or Values	Description
Clk Reverse		Normal, Invert	I Q data swap setting. Not indicative of any data received.
IF Level	± dBm		Actual IF signal level.
IF Input			Signal Presence Indicator from DCS box.
Carrier Deviation	± KHz		Actual carrier deviation from ideal specification.
GENERAL STATUS			
DCS Software			DCS software version text string.
Control		Local, Remote	Displays the unit in control; ie. Remote GPIB Avtec PTP for Windows control, or Local DCS front panel control. In Local mode, only status can be displayed at the Avtec PTP for Windows. Control can forcibly taken at either point without restriction.
Alarm			Alarm indicator. This is a red light on the front panel of the DCS, indicating that there has been a power interruption. There is no audible alarm. Alarm presence does not inhibit or alter any other functionality.
BITSYNC			
Bit sync	User Selection	1	User selection of the bit sync device within the DCS box. As of this writing, there is only one bit synchronization unit.
Mode		QPSK, BPSK	Displays the setting as reported by the DCS box. Not indicative of any data reception.
Loop Bandwidth		Narrow, Medium, Wide	Displays the allowable LBW as reported by the DCS box. Not indicative of any data reception.
Lock		Search, Lock	State of the bit synchronizer. Bit sync state is forced to Search mode when the bit sync reaches the extent of the loop bandwidth and cannot maintain lock on the input. Check specified data rate to make sure the rate is correct.
Loop Stress	percentage		Percentage indicator showing how far the VC0 has tracked up or down to maintain lock on data.
VITERBI			
Viterbi	User Selection	1	User selection of the Viterbi decoder (FEC) device within the DCS box. As of this writing, there is only one bit synchronization unit.
Decode		1 / 2	As of this writing only 1 / 2 decoding is available.
Mode		QPSK BPSK	Displays the selected data mode. Not indicative of any data reception.
Differential		On Off	Displays the selected differential decode mode. Not indicative of any data reception.

Parameter	Unit	Range or Values	Description
PRN		On Off	Displays the selected PRN (V.35 descrambler) mode. Not indicative of any data reception.
ASI		On Off	Displays the selected Alternate Symbol Inversion mode. Not indicative of any data reception.
Lock		Search, Lock	Actual Viterbi soft symbol lock status, based on data processing, as reported by the DCS box.

Module 31 DDD FORMATTER MODULE

MODULE NAME: DDDFormatter.dll

Key Functions

- Adds and/or removes encapsulation formats to/from NASCOM Blocks per user-configurable setup
- Optionally encapsulates input NASCOM Blocks into JPL-defined DDD Format
- Provides user-configurable DDD Header and Trailer fields
- Optionally Appends DDD format to any input data unit (does not have to be a NASCOM Block)
- Optionally appends Avtec PTP for Windows - defined IP Header to NASCOM Blocks that are encapsulated in DDD format
- Optionally strips DDD Header and/or Trailer from NASCOM Block data
- Optionally strips IP Header from any input data unit type
- Optionally appends IP Header to any input data unit (does not have to be a NASCOM Block encapsulated in DDD Format)

Required Hardware

- This module requires no additional hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

System limitations determine the number of DDD FORMATTER MODULES that can be added to a desktop. The total maximum allowable number of modules in a desktop is 32 and this number can not be exceeded. Since this is a software module, the CPU and data rates also determine how many of these modules can simultaneously process data.

Required Input Source, Connections, and Events

INPUT SOURCE: For data encapsulation the input data source is typically the NASCOM Blocker Module or command echoes from the Deblocker Module. For format stripping, the input is typically from a socket-type module.

CONNECTIONS: Output of this module must be connected to another module, or the data will be dropped although statistics will be maintained and reported to the status window.

EVENTS: Does not generate or expect any event input.

Module Functional Description

The DDD FORMATTER MODULE provides encapsulation and format stripping (data extraction) functions. It can either encapsulate or strip DDD formatting to/from NASCOM Blocks. It can also either append or remove the Avtec PTP for Windows-defined IP header to/from NASCOM Blocks.

The DDD format is a standard JPL DSN format typically used for transmission of NASCOM Blocks across networks. The DDD Header is 20 long and the DDD Trailer is 2 bytes. Most documentation defines the header and trailer by 16-bit words, and that standard is supported by the DDD Formatter configuration window.

The IP Header is based on the IPDU header and provides for synchronization because it includes a synchronization pattern. The IP Header is 12 bytes: 4 bytes synchronization patter, 4 bytes message length including IP header, and 4 bytes module type (defines data source module).

Both formats are defined in **Appendix B.**

Input Data Format

NASCOM Blocks are the expected input format. Depending on planned processing (stripping and/or encapsulation), the NASCOM Blocks may be encapsulated in DDD Header, DDD Trailer, and/or IP Header.

Although NASCOM Blocks are the expected format, any data unit can be input to this module.

Output Data Format

Output data format depends on module configuration (encapsulation or format stripping) and on the type of data unit input. For NASCOM Block processing, output is one of the following:

- NASCOM Blocks
- NASCOM Blocks with DDD Header
- NASCOM Blocks with DDD Header and Trailer
- NASCOM Blocks with DDD Header, DDD Trailer, and IP Header
- NASCOM Blocks with IP Header

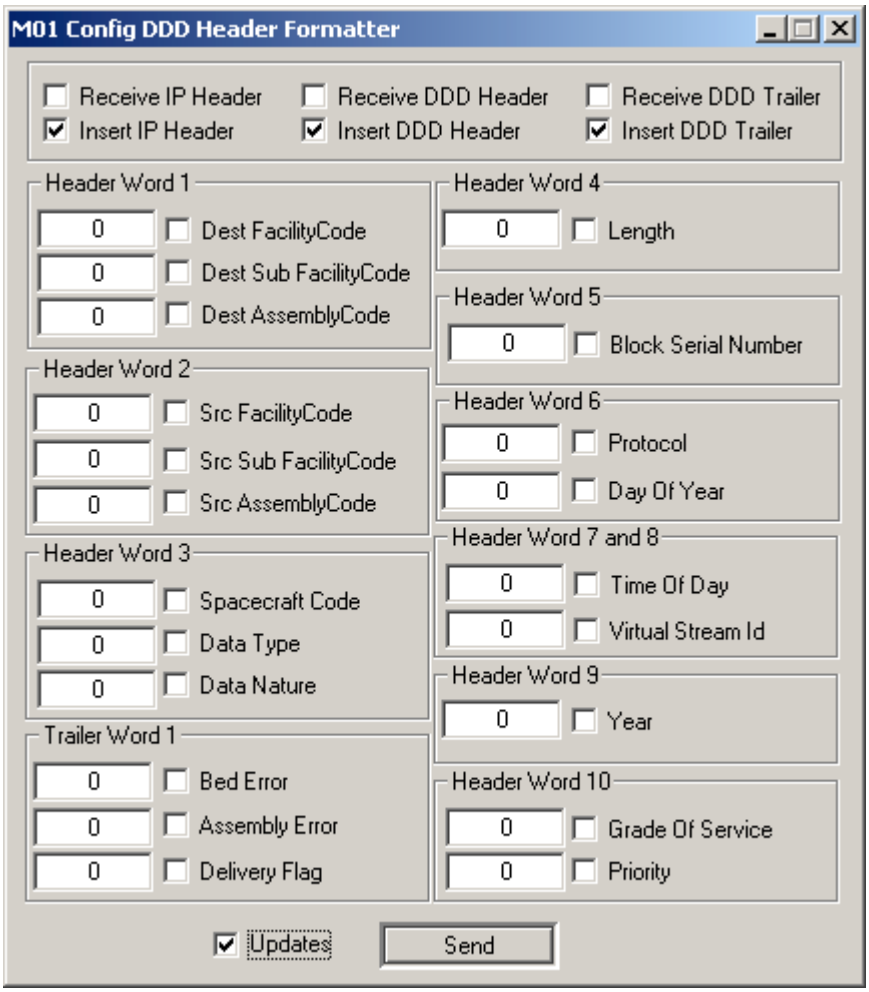
Module Window

When *DDDFormatter.dll* is added to a desktop, the DDD Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 31-1: DDD FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DDD FORMATTER MODULE Board Parameters sub-menu.



Module 31-2: Config DDD FORMATTER MODULE Window

Module 31-3: DDD FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Receive IP Header	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to receive and strip the IP Header (Avtec PTP for Windows-defined Header) from each input data unit. This is a checkbox setup: insert a check to enable and remove check to disable.

Parameter	Unit	Range or Values	Description
Receive DDD Header	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to receive and strip the DDD Header from each input data unit. The module also reads and reports DDD header information to the status window. This is a checkbox setup: insert a check to enable and remove check to disable.
Receive DDD Trailer	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to receive and strip the DDD Trailer from each input data unit. The module also reads and reports DDD Trailer information to the status window. This is a checkbox setup: insert a check to enable and remove check to disable.
Insert IP Header	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to append the IP Header (Avtec PTP for Windows-defined Header) to the beginning of each data unit input. If DDD Header is also enabled, the IP Header is appended after DDD Header is appended. The IP Header is a pre-defined format that cannot be altered by the user nor can the user configure any of the header fields with specific values. This is a checkbox setup: insert a check to enable and remove check to disable.
Insert DDD Header	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to append the DDD Header to the beginning of each data unit input. The user can set header field values using the fields provided on the window. This is a checkbox setup: insert a check to enable and remove check to disable.
Insert DDD Trailer	Selectable Setup State	Enable Disable	When enabled, this field sets the module up to append the DDD Header to the end of each data unit input. The user can set trailer field values using the fields provided on the window This is a checkbox setup: insert a check to enable and remove check to disable.
NOTE: All header and trailer word configuration fields on this window define the DDD Header and Trailer. Each word of the header and trailer is 16-bits long. The bit definitions of each word follow the DDD format. The information that is contained by each word of the DDD Format is listed on the Config DDD Formatter window. Each field of a word is defined by both a check box and an entry box. If the check box is empty, than that field is automatically filled in by the module with either a default value or a calculated value. If it is a field that can be calculated by status provided by the data source or by reading the input data, then the field is automatically filled with the value. If the module cannot perform an automatic calculation, then a default value of zero is written into that bit location. If the check box is filled, then the field is user-defined. This means that the value entered into the entry field is stamped into the DDD header/trailer of each data unit. Note: If the value entered in the console GUI or RIL is larger than the field definition, the entered value is truncated using the least significant portion of the value. In summary, each field has three possible setups: 1) User defined value that is the same for each DDD header/trailer, 2) default value of zero, or 3) an automatically inserted value that reflects the input data and its status information.			
HEADER WORD 1			
Dest Facility Code	Integer	7 bit field bits 8-14	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information.

Parameter	Unit	Range or Values	Description
Dest Sub Facility Code	Integer	4 bit field bits 4-7	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information.
Dest Assembly Code	Integer	3 bit field bits 1-3	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information..
HEADER WORD 2			
Src Facility Code	Integer	7 bit field bits 8-14	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information.
Src Sub Facility Code	Integer	4 bit field bits 4-7	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information.
Src Assembly Code	Integer	3 bit field bits 1-3	No automatic calculation. Default value is 0. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information (check box enabled).
HEADER WORD 3			
Spacecraft Code	Integer	8 bit field bits 8-15	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If the value is automatically inserted, then the value comes from one of two sources. 1) if a VC Processor module precedes the DDD Formatter at any point in the data flow, then the SCID value written to the buffer descriptor associated with the frame packed into the NASCOM Block input is read and stamped into the DDD header or 2) if no VC Processor precedes the DDD formatter in the data flow and the data unit is from the NASCOM Blocker Module, then the SCID value is read from the NASCOM Block header. A default value of zero is used for all other cases.
Data Type	Integer	7 bit field bits 1-7	May be automatic depending on check box. Provides the option to use the default value or insert a user-defined value into the DDD Header at the bits assigned to this information.
Data Nature	Integer	1 bit field bit 0 value is either 0 or 1	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. This flag was defined to indicate either playback or real time data. If it is set up to automatically calculate the value and data is input from a File Playback Module, then this bit gets set to one. For all other cases, the bit is automatically set to zero.
HEADER WORD 4			

Parameter	Unit	Range or Values	Description
Length	Integer	16 bit field	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If set up to automatically calculate, this field represents the total output data length excluding IP Header: length of input data unit + DDD Header + DDD Trailer.
HEADER WORD 5			
Block Serial Number	Integer	16 bit field	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. When set to auto, it assigns a block serial number to each input block starting at 0 and incrementing by 1 for each consecutive block. Roll over is at 16 bits.
HEADER WORD 6			
Protocol	Integer	6-bit field bits 10-15	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If this field is set to auto, it defines the input data as blocked or unblocked: 1= unblocked and 2= blocked. Two cases cause the data to be flagged as blocked: 1) the data source module is the NASCOM Deblocker Module and the data length is 600 bytes (which indicates it is a command echo) or 2) if the source module is the NASCOM Blocker Module. For all other cases, it will automatically be identified as unblocked data (field value is 1).
Day of Year	Integer day of the year	10-bit field: bits 8-9: hundreds digit bits 4-7: tens digit bits 0-3: units digit 0 to 366	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If auto, then day of year is based on buffer descriptor information associated with the input data. The buffer descriptor information will vary depending on the data flow and the modules that precede the DDD Formatter. A summary of possible time/day sources follow: 1) If the Serial Input Module precedes this module in the data flow, then the ERT/PB4 time recorded by the Serial In module is read from a buffer descriptor 2) If the data source is the NASCOM Deblocker Module, then time is read from the NASCOM Block header 3) If the data source is the TFDH Module (socket), then the time provided by this module's buffer descriptor is used. 4) If the data source is either the Playback or Spooler Modules and data was recorded with MUX header and PB4 time stamp header, then the time from the time stamp is used. 5) If the data source is either the Playback or Spooler Modules and data was not recorded with time stamp header, then the time will be whatever time it was pulled off the recorder/spooler.
HEADER WORD 7 AND 8			

Parameter	Unit	Range or Values	Description
Time of Day	Integer milli-seconds since midnight	24 bit field Word 7: bits 0-15 Word 8: bits 8-15	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If auto, then day of year is based on buffer descriptor information associated with the input data. The buffer descriptor information will vary depending on the data flow and the modules that precede the DDD Formatter. A summary of possible time/day sources is provided in the description of Day of Year for word 6. Static value (user-defined) can be used for testing
Virtual Stream ID	Integer	8-bit field Word 8: bits 0-7	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. If set to auto, then one of two cases apply: 1) If the VC Processor Module precedes the DDD Formatter module in the data flow, then the VCID value of the frame (buffer descriptor from VC Processor) is used for this field. 2) If the source is non-CCSDS (no VC processor) then this field is et to the input source module number + 100. This means that a value of 100 or higher is a telemetry stream and not CCSDS.
HEADER WORD 9			
Year	Integer	4-bits each: thousands digit, hundreds digit, tens digit, units digit	This field can be automatically calculated (check box disabled) or the field can be user-defined (check box enabled), which means the user-defined value is stamped into every DDD Header. For auto, the year is calculated based on the time source. The same criteria for time source based on data source used as defined in word six, Day of Year.
HEADER WORD 10			
Grade of Service	Integer	8 bit field bits 8-15	Provides the option to use the default value (0) or insert a user-defined value into the DDD Header at the bits assigned to this information.
Priority	Integer	2 bit field bits 5-6	Provides the option to use the default value (0) or insert a user-defined value into the DDD Header at the bits assigned to this information.
TRAILER WORD 1			
Bed Error	Integer	1 bit field bit 0	Provides the option to use the default value (0) or insert a user-defined value into the DDD Header at the bits assigned to this information.
Assembly Error	Integer	1 bit field bit 1	Provides the option to use the default value (0) or insert a user-defined value into the DDD Header at the bits assigned to this information.
Delivery Flag	Integer	1 bit field bit 2	Provides the option to use the default value (0) or insert a user-defined value into the DDD Header at the bits assigned to this information.

Extended Options Window Definition

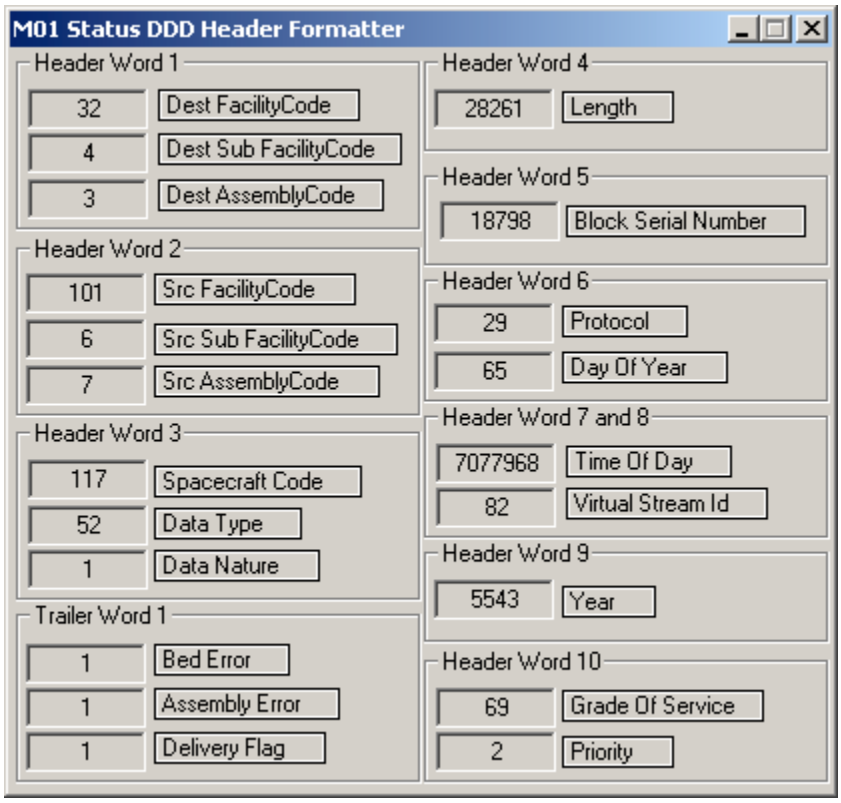
There is no extended options window for the DDD FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the DDD FORMATTER MODULE.

Status Window Definition

When you click the Status button on the DDD FORMATTER MODULE window, a Status DDD FORMATTER window appears, as shown below.



Module 31-4: Status DDD FORMATTER MODULE Window

Module 31-5: DDD FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
HEADER WORD 1			
Dest Facility Code	Integer	7 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.

Parameter	Unit	Range or Values	Description
Dest Sub Facility Code	Integer	4 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Dest Assembly Code	Integer	3 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 2			
Src Facility Code	Integer	7 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Src Sub Facility code	Integer	4 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Src Assembly Code	Integer	3 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 3			
Spacecraft Code	Integer	8 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Data Type	Integer	7 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Data Nature	Integer	1 bit field value is either 0 or 1	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 4			
Length	Integer	16 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 5			
Block Serial Number	Integer	16 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 6			
Protocol	Integer	6-bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.

Parameter	Unit	Range or Values	Description
Day of Year	Integer day of the year	10-bit field: 2 bits hundreds digit, 4 bits tens digit, 4 bits units digit 0 to 366	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 7 AND 8			
Time of Day	Integer milliseconds since midnight	24 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Virtual Stream ID	Integer	8-bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 9			
Year	Integer	4-bits each: thousands digit, hundreds digit, tens digit, units digit	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
HEADER WORD 10			
Grade of Service	Integer	8 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
Priority	Integer	2 bit field	Reports the value read from the DDD Header at the bit location assigned to this field for the current DDD Header input to the module.
TRAILER WORD 1			
Bed Error	Integer	1 bit field	Reports the value read from the DDD Trailer at the bit location assigned to this field for the current DDD Trailer input to the module.
Assembly Error	Integer	1 bit field	Reports the value read from the DDD Trailer at the bit location assigned to this field for the current DDD Trailer input to the module.
Delivery Flag	Integer	1 bit field	Reports the value read from the DDD Trailer at the bit location assigned to this field for the current DDD Trailer input to the module.

Module 32 DECOMMUTATOR MODULE

MODULE NAME: Decom.dll

Key Functions

- User defined display of TDM, CCSDS parameters. Value can be made up of as many as four noncontiguous pieces, each up to 8 bits, from the same or different frame buffer and individually assembled into a usable parameter up to 32 bits in length.
- Maximum / Minimum / Average processing on defined parameters
- Engineering Unit Calibration using seventh order polynomial calculation or linear interpolation using slope / intercept calculation
- Online help for the decom module
- Analog saturation detection and flagging
- Supercom and subcom data definition processing
- Discrete State and Analog Range Mnemonic Labeling
- Red / Yellow, High / Low limit checking and flagging of parameters. Current and latched limit status reporting. User activated latch release. Limits are individually selectable, and can be inclusive or exclusive, standard or inverted.
- User defined display of any parameter within a buffer, even if data is not telemetry.
- Conversion to decimal, octal, binary, hexadecimal, engineering unit, discrete state, analog range mnemonic, IEEE fp, scientific notation, BCD, signed or unsigned, ASCII text.
- Color calibration for the visually challenged
- User defined Delta Limit Checking and flagging
- Data Inversion and Bit Reversal (LSB first) processing.

Required Hardware

- None

Add Module Guidelines

There are no iteration quantity restrictions. One module could send data to several different decom modules simultaneously, e.g. to feed a C&DH decom window and a power subsystem window, etc.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives data buffers sent from other modules.

CONNECTIONS: The module sending data must be connected to the decom module.

EVENTS: No events are expected or generated.

Module Functional Description

This module receives data passed to it by other modules. There is no restriction on buffer length; any valid Avtec PTP for Windows buffer will work. As of this writing, the maximum size Avtec PTP for Windows buffer is 16384 bytes.

The Config window is used to define up to sixteen data points (per iteration of this module) to be extracted from the data and displayed on the console gui. Each data point (parameter) may be from 1 to 32 bits in length, and the source bits need not be contiguous in the buffer. To maximize screen space, each parameter is defined in the same entry fields within the "Parameter" area of the Config window. The specific parameter that is being worked is selected by the listbox at the top of the "Parameter" area. The data in all the fields will change as the user cycles through this listbox. The "Components", "Descriptors", "Polynomial", and "LineSegments" areas within the "Parameter" box operate in the same manner; they accept multiple inputs in each field for each parameter. The "Send" button is pressed to record each individual input in these multi-use fields. The user may need to push the "Send" button up to thirty times per parameter definition in order to complete the most complex of definitions where all available input is required. Typically, only four or five presses of the "Send" button are necessary to define most parameters.

Two general modes of decommutation exist. In the first case, a counter is defined in the input stream to allow identification of the data in the buffer. A TDM frame counter or a CCSDS VC id fits this purpose, although any data could be used depending on the users' requirements to identify the content of the buffer. This counter must occur in every buffer. It may be up to 8 bits or 9 to 16 bits long. Specify the size in the "Gross Size" parameter listbox. The start byte of the counter (the most significant byte is numbered 1) is specified in decimal in the "Byte Location" field. This byte (or two consecutive bytes, if the size is set for 9 to 16 bits) is extracted from the incoming data. A hexadecimal mask, "Mask" is applied to and-out any unrelated bits. A "Shift" can be chosen to position the bits correctly to achieve the desired value range for the counter. If the counter bits are revered in the stream, checking the "BR" (bit reverse) box will re-order the bits after extraction to be most significant bit first. If the counter bits are inverted, check the "1sC" box to one's complement the bits after extraction. Select the required shift function, either no shift, or left / right 1 to 7 bits. The behavior of the counter is also defined here. The user should specify whether the counter increments or decrements, the minimum value "From", the final value, "To" and the size of each of the steps, "By". From, To, and By are decimal values. This behavior capability allows the software to check for sequence errors in the data, and alert the user to marginal quality or missing data. The other general mode of decommutation is defined as "No Counter" in the "Gross Size" listbox. This causes all parts of all data points (parameters) to be extracted from each buffer. No sequence checking can be preformed in this mode.

Parameters are defined, one at a time, in the "Parameter box". Parameters are assigned relative index numbers, from 1 to 16. All indices possess all capabilities; thus any parameter can be defined in any of the sixteen indices. It is often helpful to the user to group the parameters in some manner that makes the Status window easier to read. Consider grouping parameters of like type together, or group all the parameters together that apply to, for

example, battery 1; followed by all the parameters that apply to battery 2, or whatever makes the Status window easier to monitor. The parameter number is selected in a listbox at the very top of the parameter box. All values displayed in the parameter box apply to the parameter number selected in this listbox. The selected parameter definition may be cleared by pressing the "Clear Parameter Definition" button when the desired parameter number is displayed. The "Clear Parameter Definition" button will reset all definitions for "Components", "Limits", "Descriptors", "Polynomials", and "Line Segments" to their initial states.

The number of bits in the parameter value is defined in an entry field box. A checkbox next to the entry field is used to enable or disable the decommutation of this parameter. A description of the parameter, such as "Battery 1 Voltage" may be entered in the "Desc" field. The color of the description is entered in the Red / Green / Blue boxes to the right of the description. Red / Green / Blue is defined as three decimal numbers from 0 to 255. A value of zero turns the specific color gun off completely, a value of 255 selects full on. An RGB value of 0/0/0 results in the darkest black; an RGB value of 255/255/255 results in the brightest white. Consider changing the color of each grouping of parameters to invite better readability. If the value of the parameter needs to be inverted after decommutation, select the "Ones Complement" checkbox. This box causes the value to be bit-wise inverted prior to storage in the display buffer. If the value of a parameter is extracted from the input buffer in the least-significant-bit-first format, check the "Bit Reverse" box to reverse the order of the bits. Limit checking, calibration, and descriptor ranging is done after any bit reversal or inversion.

For test purposes, the value of a parameter may be forced to a user specified value. The value may be specified in hexadecimal, octal, or decimal; when the display window updates, the hexadecimal value will be displayed. Check the "Force Value to" box and enter a value in the entry field, and select the desired base. This value will be bit-wise anded with a mask representative of the number of bits in the value, to insure that the value forced does not exceed the defined size of the parameter. When the Force Value box is checked, the decommutated value is discarded in favor of the entered value. The value is then processed by the limit checking, calibration, and descriptor software just as if the user supplied value was a decommutated value. The forced value will continue to be processed until the "Force Value To" checkbox is cleared. A flag "f" is always included in the status display indicating that the value was forced, so that the user will not confuse the forced value with real data. The force value capability is useful for validating limit and calibration definitions.

Pushing the button labeled "?" displays a general help screen for this module.

The user defines up to four pieces (components) of the value. These components may be up to eight bits in length, and extracted from any byte in the buffer, or from any other buffer to be received. Components are defined one at a time for the parameter. To simplify the functionality and understand how components fit into a parameter, imagine a picture of the parameter, with the extracted byte right-justified with the least significant bit of the parameter; assume the following example:

```
                    E E E E E E E E     (8 bit extraction)

P P P P P P P P P P P P P P P P     (16 bit parameter)
```

The byte number in the input buffer is defined in the entry field "Byte". This value is relative to 1, meaning that the most significant byte in the input buffer is numbered 1. Care should be taken so as not to specify a byte number greater than the length of the input buffer that is being used. The eight bits are shown above in the example with capital letter "E"s. Picture them initially aligned with the parameter at the least significant bit, as shown. Enter a hexadecimal mask in the "Mask" box to AND-out bits which are unrelated to this parameter:

```
            0 E E E E E E 0           (6 desired bits)
```

would result in a mask value of 7E; to be entered in the "Mask" box. Enter the numerically smallest value of the counter (frame counter, VC id, etc) when this extraction is first to take place in the "Counter" field. If a value appears in every buffer, enter 0 for the Counter. In the "No Counter" mode, this box is grayed.

Lastly, position the remaining bits as they are required to appear in the parameter. The "<<" specifies the number of bits to shift the extracted byte (the E's in the above example) to the left. Imagine them physically above their correct position in the parameter. In the picture below, the bits have been shifted three bits to the left:

```
            0 E E E E E E 0           (6 desired bits)

P P P P P P P P P P P P P P P P     (16 bit parameter)
```

Assume that the extracted bits (E's) are now physically above their destination bits in the parameter (P's) bits. Recall that the undesired bits in the extracted bits (E's) have already been set to binary zeroes by the masking that was done prior to shifting. Bits that were selected by the mask in the "Mask" box will be inserted into the parameter. Bits anded-out by the mask will remain unchanged in the parameter value:

```
            0 E E E E E E 0           (6 desired bits)

P P P P P P E E E E E E P P P P     (16 bit parameter)
```

In the above example, the six desired bits have been positioned and inserted into the sixteenth bit parameter. These six desired bits were inserted regardless of the previous content of those six bit positions in the sixteenth bit parameter. The other bits in the parameter (P's) remain unchanged and need to be filled in by another of the four possible components of the parameter value. The user should ensure that all bits in the parameter are accounted for in the components section. A simple error in the components definition can result in strange, unexpected values being processed and displayed for a parameter.

A second example shows what would happen if the user had elected to shift the extracted bits one bit to the right instead of three bits to the left:

```
                  0 E E E E E E 0   (6 desired bits)
```

```
P P P P P P P P P P E E E E E E     (16 bit parameter)
```

Only one of the left shift ("<<") or right shift (">>") boxes should be non-zero. Enter zero for the other box. If both shift boxes are zeroes, no shift takes place. If a non-zero value does happen to exist for a component, the specified left shift is preformed first, and then the specified right shift. Bits shifted out of either end above parameter are lost. Binary zeroes are shifted in as required.

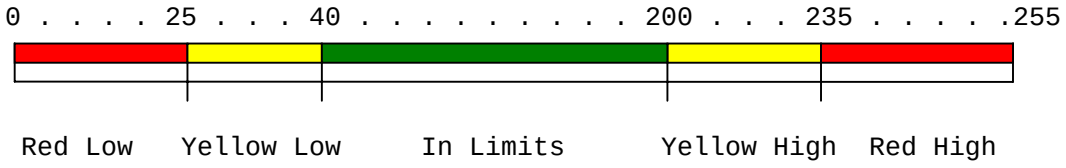
If this component of the parameter is subcommed, a box labeled "Subcom Repeat" allows the user to define the depth of the subcom. For example, entering the value 4 in the subcom repeat box would cause this component to be extracted when the counter in the data is equal to the specified counter and every fourth value thereafter (modulo 4). Suppose the counter value specified in the component was 0, and the subcom repeat was set to four. This would cause the component to be extracted when the counter extracted from the buffer was equal to 0, 4, 8, 12, 16,…etc. Alternatively, if the counter value specified in the component was 10, and the subcom repeat was set to 3, the component bits would be extracted and placed into the parameter value when the counter extracted from the buffer was equal to 10, 13, 16, 19, 22, …etc. A subcom repeat value of 1 causes the component to be extracted from all buffers whose counter in the data values are equal to or greater than the component specified counter value. A subcom repeat value of 0 disables any repeating, and causes the component to be extracted only when the value of the counter in the data is exactly equal to the entered component counter. In the "No Counter" mode, this box is grayed.

Be sure to enable the component by selecting the checkbox in the front of the positioning boxes. As each component of the parameter is defined, push the send button to record it, and then the second, third, or fourth component can be entered in the same boxes. Changes may be discarded by pressing the "Updates" checkbox.

Limit checking may be used to validate the expected range of the parameters. Values are limit checked each time a new sample has been extracted from the input data. Note that the value of the parameter is limit checked, not the values of the individual components making up that value. Only binary, uncalibrated values are limit checked. The current limit state is always displayed with the current value of the parameter. A latched limit status is also displayed, showing whether or not the value has been out of limits since the last user limit status acknowledgement. This allows the user to know if the value went out of limits while he was looking at something else, or if the parameter went out of limits and went back in limits between window updates. If a yellow limits violation has been latched, and subsequently a red limits violation occurs, the latch is updated with the red violation. This re-latching-before-reset mechanism allows the user to recognize the worst-case limit violation that has occurred. Once latched, the status is held until manually reset using the "Reset Latched Flags" button on the Status window. Maximum / Minimum / Average history is kept on the value of all parameters and it available at the push of the "MaxMin" button on the status page.

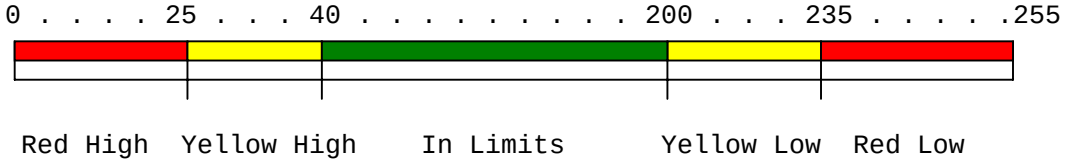
Red / Yellow limits are defined in the Limits box. Three possible red / yellow limit checking strategies are available. The first, "RL < YL < in < YH < RH", supplies the typical limits

checking. Suppose, for example, an 8 bit parameter (values 0 to 255 decimal) to be limit checked as follows:



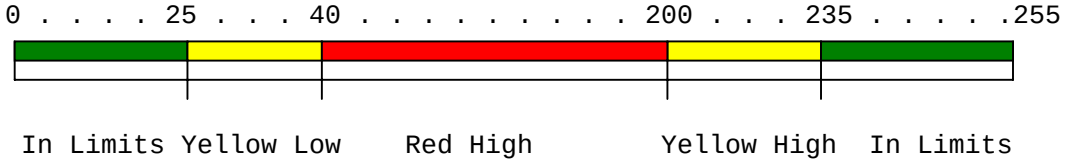
This selection would display the value in red, yellow or green, flagged with "RL", "YL", "in", "YH", or "RH" based on the current value of the parameter. This setting is called "inclusive" limits, because the part of the scale that is in limits is numerically between the part that is outside limits.

The second red / yellow strategy is the same as the first, except that the high and low flags are reversed before they are reported. This strategy is useful for thermisters or other parameters whose calibration slope is negative. This selection reports out of limits highs when the value is low, and out of limits lows when the value is high. Pictorially:



To use this option select "RH < YH < in < YL < RL" in the limits listbox.

The third red / yellow strategy is called exclusive limit checking. The expected "in limits" range wraps around the 0 / 255 saturation end point. Pictorially, this can be shown as follows:



The expected range in this example is from 235 to 25 wrapping thru 255 and 0. Note that in this selection, there is no red low setting, since the active red area is in the middle of the scale.

Once a limit checking strategy is selected, enter the limits values in the four boxes which represent their respective state above them in the listbox. Each limit can be selectively enabled or disabled at will, using the checkboxes below the limit values. To turn off limit checking for a parameter, clear the checks in the four checkboxes. There is no need to clear the defined limit values.

Delta limit checking can be defined using the "Delta" checkbox and entry field. Delta limits alert the user that the value of a parameter has changed by more than a certain amount. Enter

the delta limit and enable the checkbox (and press "Send") to set a delta limit. The new value of the parameter is compared with the previous value. If the absolute difference between the previous value and the current value exceeds the specified delta limit value, a delta limit violation has occurred. The status page will show a "D" flag if the parameter exceeds the delta limit. The value will change to blue, unless the parameters' color has been preempted by a red / yellow limit violation. If the parameter is decommutated faster than the update rate of the Status window, the user may not see a value that is only flagged with a delta violation.

Saturation checking can be enabled to check the value of a parameter. This usually applies only to an analog measurement. If the value falls to all binary zeroes or rises to all binary ones, the value will be flagged with the saturation flag ("*") telling the user that the displayed value is saturated and can go no lower (or higher). This may seem obvious if the user is displaying the value in binary or decimal, but is not readily apparent when the value is calibrated in engineering units prior to display. If the parameter is decommutated faster that the update rate of the Status window, the user may not see a value that is flagged with the saturation indicator.

Some users have trouble seeing a variety of colors. Each color can be adjusted to suit the individuals' requirements for improved viewing or for printing screen content. Press the "Tweak Colors" button for a window which allows the user to change the appearance of red, green, blue, yellow, black, white, and the screen background colors. Each color is specified using decimal values in the 0 to 255 range for the Red, Green, and Blue color guns. A value of 0,0,0 (RGB) results in the darkest black color. A value of 255,255,255 (RGB) results in the brightest white available. A sample of the modified color is available on the "Extended Config" window. Use the regular "Send" button on the Config window to activate the changes.

The definition of descriptors for parameters can translate bit patterns or analog ranges to a specific word in a specific color. This can provide an "all's well" or a trouble indication that is more readily apparent that a specific value. The user may define up to eight ranges or bit patterns to translate to descriptor form for each parameter. Each parameter is defined using the same entry fields in the gui. The descriptor number selects which of the eight descriptors is being defined. Press "Send" to record each entry as it is defined. To view a descriptor definition entry, set the descriptor list box to the desired entry index.

The user enters a set of low and high range of values for each parameter. A bit pattern is treated as if it was an analog value. To active only a specific pattern, set the Low and High value to the same value; one that is representative of the unsigned decimal equivalent of that bit pattern. The maximum value that can be entered for the low or high range check is 65535. Enter a text of seven or less characters to be displayed as the value of the parameter when discrete state display is chosen for that parameter on the status page. Select a color for the text using RGB values in the rage of 0 to 255. This RGB selection operates the same as the Tweak Colors definition or the parameter description text color. Each state can be individually turned on or off as desired. The ranges are checked in order starting with the first entry. A second entry with a range that includes part or all of a previously defined range

will not be found. When the value of the parameter is not within any of the ranges, the value of the parameter will be displayed in decimal in the color red.

Each parameter may be calibrated into engineering units (volts, amps, watts, etc.) using either (or both) of two different strategies. When a polynomial calibration curve or line segment slope / intercept approximation is defined, the calibrated value can be displayed on the status window. The user has a choice of the number of significant decimal places in the floating point result, and an optional forced plus sign when the value is positive. Units of measure are defined in the Engineering Unit Calibration Enable box. Units are displayed in the same color as the description of the parameter entered earlier. The units of measure field is simply text; i.e. no internal processes depend on this entry, and no other conversions take place; e.g. degrees to radians, or other such functions.

Polynomial calibration requires coefficients to solve an equation. Up to seventh order polynomial calibration can be accommodated. This means that up to eight floating point coefficients are required. The user should enter 0.0 for any unused coefficients. Coefficients are numbered 0 to 7. Each is entered in the same entry field, and can be reviewed or changed individually, at will. Press the "Send" button to record each as it is entered. If the user enters an integer type number, it will be automatically converted to floating point. All coefficients are displayed in signed scientific notation, but may be entered in any normal C++ integer or floating number format. The equation solved using the coefficients is:

$$EU = C_0 + (C_1 \times V^1) + (C_2 \times V^2) + (C_3 \times V^3) + (C_4 \times V^4) + (C_5 \times V^5) + (C_6 \times V^6) + (C_7 \times V^7)$$

where:

C_n	is the defined coefficient;
V^n	is the current raw binary value of the parameter;
EU	is the calibrated value.

The polynomial calibration is enabled using the "Polynomial" checkbox in the "Engineering Unit Calibration Enable" section of the gui.

Linear Interpolation Calibration is also available. An engineering unit value is approximated using several points to form graph containing line segments (raw vs. EU). Up to eight points can be specified, resulting in a maximum of seven line segments, each of a given slope. An intercept is formed along one of the lines to calculate the approximate engineering unit value. Raw values to be calibrated, which occur numerically either before the first point or after the last cause the appropriate (first or last) segment to be extended at its' same slope to allow a calibration to proceed. Points must be defined in ascending order, with no duplicates, regardless or whether a positive or negative slope is to be formed. The user enters a raw value, followed by a calibrated value for that point. This is repeated until all desired points (up to 8) are entered.

Calibration is preformed by checking each endpoint of each (up to 7) of the lines formed by the entered points. When a line segment with a range containing the current raw value of the parameter is found, that segment is used for calibration. If the parameter value is equal to either endpoint, the calibrated value of the endpoint (entered by the user) is deemed the calibrated value. If the raw value of the parameter is found along one of the line segments, the following equations are solved:

$$\text{Slope} = (L_{ep} - L_{bp}) \div (C_{ep} - C_{bp})$$

$$\text{Intercept} = L_{bp}$$

$$\text{EU} = ((V - L_{bp}) \times \text{Slope}) + \text{Intercept}$$

where:

L_{ep}	is the end point of the chosen line segment;
L_{bp}	is the begin point of the chosen line segment;
C_{ep}	is the calibrated value of the end point of the line;
C_{bp}	is the calibrated value of the begin point of the line;
EU	is the resultant calibrated value.

Each point is entered one at a time in the same entry boxes on the gui. Press "Send" to record each point. Press "Updates" to discard your current input. Any of the points can be reviewed and changed individually. There is an enable checkbox for each point. A minimum of two points must be defined and enabled to form one line. Calibration is enabled by the "Line Segment" checkbox in the "Engineering Unit Calibration" section of the gui.

The Status window is used to view the data display. Raw values are also available through the Remote Interface Library (RIL).

The status window displays the current counter value extracted from the data. It is green while it is changing due to data. If no change occurs in five seconds, the value turns black. The counter may be displayed in decimal, octal or hexadecimal, and may be changed back and forth as desired. The number of times the counter has returned to its begin point is displayed as Rollovers. These could be called major frames in the TDM world. Sequence errors of the counter, as defined in the From / To / By section of the Config window are reported. The PB4 time attached to the Avtec PTP for Windows buffer is displayed.

The main part of the Status window consists of sixteen rows of data, one for each parameter. The parameter description, value, units of measure and flags (current limits and latched limits) are displayed. The latched flags can be acknowledged by the user at any time by pressing the "Reset Latched Flags" button. This function sets the latched flags to the same status as the current flags. When a parameter goes out of limits, the latched flags column will retain the flag, either yellow or red, until the user acknowledges that he understands. This

latching mechanism handles the situation where the user looks away for a few seconds or when the input data is fast enough that every sample cannot be displayed. Each sample will still be limit checked, and if it falls out yellow or red, the flag will be retained until the user clears it.

The "MaxMin" window displays the initial value, the minimum value, the maximum value, the number of samples and the simple average of all values.

The last received value and current flags are always displayed in the Status window. The initial, minimum, and maximum values include the flags that were displayed at the time of occurrence. The values are not re-limit checked or re-calibrated if changes are made to the parameters' limit or calibration definition after the maximum or minimum value is encountered. The Maxmin window may be reset at any time by pressing the "MaxMin Init" button on the Status page.

Values may be displayed in binary, discrete descriptor, engineering units (polynomial or linear interpolation calibrated; normal or scientific notation), octal, BCD, decimal, signed decimal, IEEE floating point (normal or scientific), hexadecimal, or ASCII text. The user may change the base of each parameter individually at will. The display will automatically change as the user scrolls through the listbox of bases. When the user changes the base of a parameter on the Status window, the corresponding base of that parameter in the MaxMin window will change automatically.

The user can enter a forty character window title in the entry box at the bottom of the Config window. This text is displayed in the title bar of the Status window. The users' text is appended to the standard Avtec PTP for Windows title bar text which identifies the module. The text is appended when the Status window is opened; subsequent changes require that the Status window be closed and re-opened to display the changed title bar text.

In a future Avtec PTP for Windows release it is planned to export the data in a manner acceptable to off the shelf strip charting and plotting packages.

Input Data Format

The user defines the format of the incoming data.

Output Data Format

Console display and Remote Interface Library values. Data output to another subsystem is planned in a future TBD release.

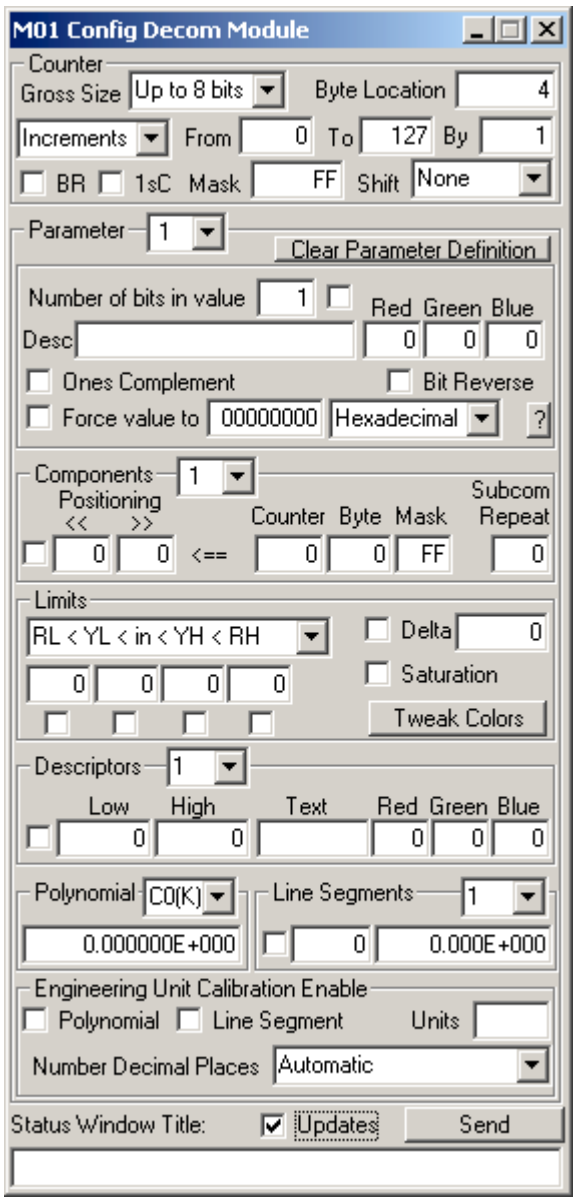
Module Window

When Decom.dll is added to a desktop, the Decom dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 32-1: DECOM MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DECOM MODULE Board Parameters sub-menu.



Module 32-2: Config DECOM MODULE Window

Module 32-3: DECOM MODULE Configure Window Parameters

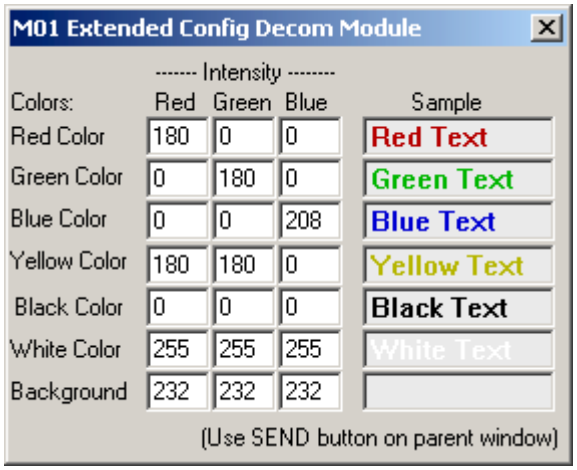
Parameter	Unit	Range or Values	Description
COUNTER			
Gross Size	Listbox selection	No counter, Up to 8 bits, 9 to 16 bits	Defines the size of the counter (part of the data buffer which allows the user to dissect the remainder of the buffer); also disables the counter function.
Byte Location	Decimal	1 to 16384	The byte number of the counter (starting byte of two consecutive bytes if 9 to 16 bits is chosen for Gross Size.
Increment/ Decrement	Listbox selection	Increments Decrements	Behavior of the counter
BR	Checkbox		If checked, causes the counter bits to be reversed in order.
1sC	Checkbox		If checked, causes the counter bits to be one's complemented (inverted)
Mask	Hexa-decimal	00 to FF 0000 to FFFF	Hexadecimal mask to remove unrelated bits from the extracted counter. Could be 00 to FF is Up to 8 bits is selected for Gross Size; 0000 to FFFF if 9 to 16 bits is selected for Gross Size.
Shift	Listbox selection	None, Left 1 bit to Left 7 bits, Right 1 bit to Right 7 bits	Positioning action to be taken on the extracted counter value.
PARAMETER			
Parameter	Listbox selection	1 to 16	Selects which parameter is to be acted upon.
Clear Parameter Definition	Button	Press	Clears a definition for a parameter completely.
PARAMETER/GENERAL			
Number of bits in value	Decimal	1 to 32	Size of parameter in bits.
Desc	Text	Up to 32 chars	Descriptive text for the parameter, e.g. "Battery 1 Voltage"
Ones Complement	Checkbox		If checked, parameter is One's Complemented (inverted) prior to limit check, calibration, display.
Bit Reverse	Checkbox		If checked, parameter is Bit Order Reversed prior to limit check, calibration, display.
Force Value to	Hex, Octal, or Decimal	Value acceptable according to base	Value to replace the decommutated value for test purposes.
?	Button	Press	Displays a help screen for this module.
PARAMETER/COMPONENTS			

Parameter	Unit	Range or Values	Description
Components	Listbox selection	1 to 4	Component number of this parameter to be acted upon
Positioning	Decimal Decimal	Left 0 to 7 Right 0 to 7	Positions the extracted data bit-wise within the parameter.
Counter/ Byte/Mask	Decimal Decimal Hexa-decimal	0 to 65535 0 to 16384 00 to FF	Counter value at which to decommutate this component of a parameter Byte number, relative to 1, that contains the data to be decommutated for this component of a parameter Mask to be applied to the extracted byte (component)
Subcom Repeat	Decimal	0 to 32767	Depth of subcom. 0 means no subcom repeat. 1 means repeat every buffer after the specified counter. Other value specifies n^{th} counter repeat.
PARAMETER/LIMITS			
Limits	Listbox selection	RL<YL<in<YH<RH RH<YH<in<YL<RL in<YL<RH<YH<in	Specifies the limit checking strategy. Standard Inclusive / Inverted Inclusive / Exclusive
Delta	Checkbox Decimal	Press 0 to 65535	Specifies delta limit enable / disable and value. If checked, the current value is compared with the previous value to see if the value changed by more than the specified value.
Saturation	Checkbox		If checked, the value of the parameter is compared to all binary zeroes and all binary ones. Saturation has occurred if either is true.
Tweak Colors/ <u>Extended Config Window</u>	Button	Press	Display the Extended Config Window to Tweak Colors. The actual displayed color can be changed to accommodate personal preference or visual requirements.
PARAMETER/DESCRIPTORS			
Descriptors	Listbox Selection	1 to 8	Specifies the index of the descriptor entry to be acted upon.
Low	Decimal	0 to 65535	Low value or bit pattern for this text descriptor entry
High	Decimal	0 to 65535	High value or bit pattern for this text descriptor entry
Text	Text	Up to 7 chars	Actual text to be displayed in the value field when descriptor is the chosen display format.
Red	Decimal	0 to 255	Red gun value; 0 for lowest gain, 255 for highest gain
Green	Decimal	0 to 255	Green gun value
Blue	Decimal	0 to 255	Blue gun value
PARAMETER/POLYNOMIAL			
Polynomial	Listbox Selection	C0(K), C1 to C7	Selects which coefficient is currently being entered or viewed for this parameter.

Parameter	Unit	Range or Values	Description
Entry field	Float	Any float or integer	Entry box for each coefficient for this parameter. C0 is the constant (K).
PARAMETER/LINE SEGMENTS			
Line Segments	Listbox Selection	1 to 8	Selects which point entry is currently being set or viewed for this parameter.
Entry Field	Decimal Float/ Decimal		Specifies a point (raw value, X-axis) and a corresponding engineering value (float, Y-axis) for this point.
PARAMETER/ENGINEERING UNIT CALIBRATION ENABLE			
Polynomial	Checkbox		Enables polynomial calibration for the selected parameter
Line Segment	Checkbox		Enables linear interpolation calibration for this parameter.
Units	Text	Up to 7 chars	Units of Measure text for the selected parameter; color is the same as the description of the parameter
Number Decimal Places	Listbox Selection	None, Automatic, 1 to 7, None forced+sign, Auto forced+sign, 1 to 7 forced+sign	Number of decimal places to display. Optional '+' can be displayed when the value is positive ('-' is always displayed if the value is negative). Automatic is the default based on the accuracy of the computer that this module is being run upon.
OTHER SETUP			
Status Window Title	Text	Up to 40 chars	Title of Status Window. Useful when several Decom Module Status window are being displayed simultaneously. Window title is only updated when opened. The entered text is appended to the standard Avtec PTP for Windows window title that identifies the module. If the text has been changed in the Config Window, the Status window will have to be closed and re-opened to see the change.

Extended Options Window Definition

The Extended Config window is displayed when the Tweak Colors option of the Parameters/Components setup is selected.



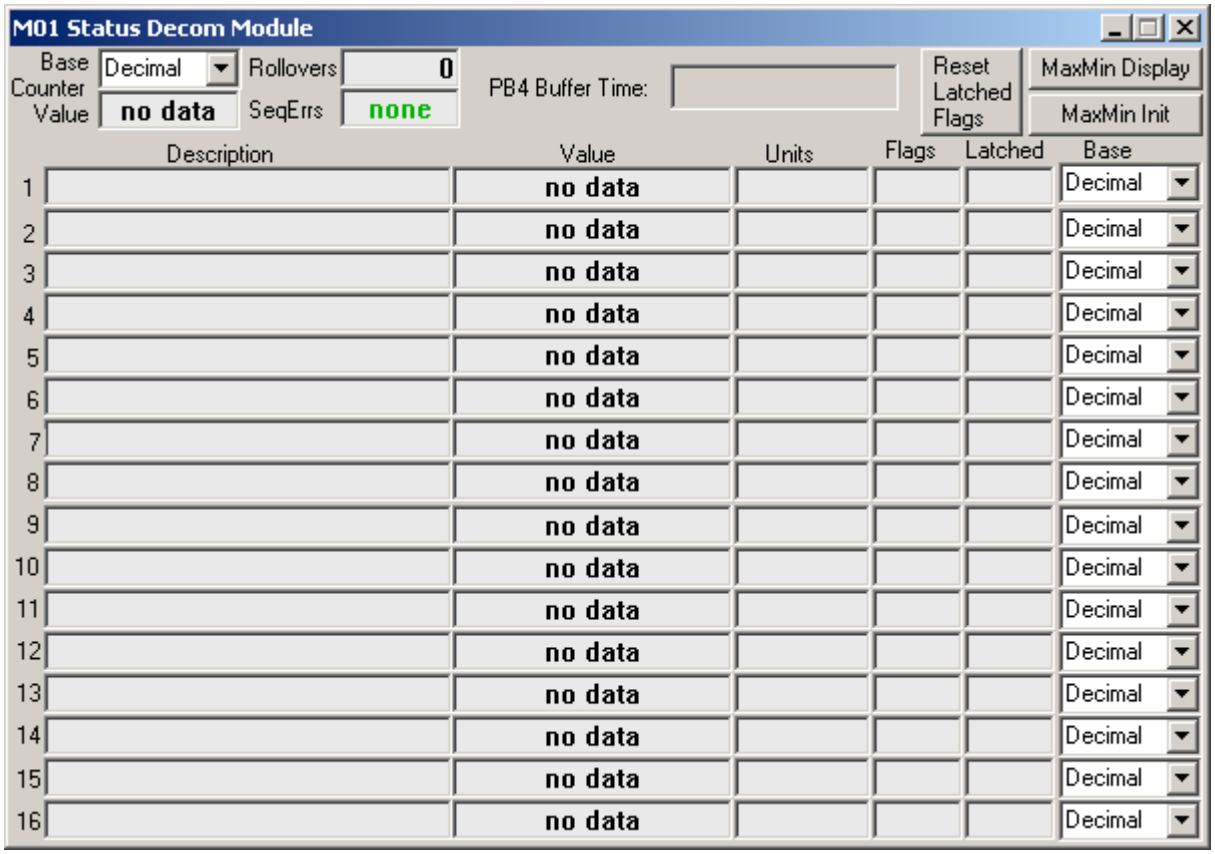
Module 32-4: Config DECOM MODULE Tweak Colors Window

Configuration Notes

There are no configuration notes for the DECOM MODULE.

Status Window Definition

When you click the Status button on the DECOM MODULE window, a Status DECOM window appears, as shown below. Pressing on the MaxMin Display button will bring up the Extended Status Decom window.



Module 32-5: Status DECOM MODULE Window

M01 Extended Status Decom Module

	Initial Value	Minimum Value	Maximum Value	Number Samples	Simple Average
1	no data	no data	no data	0	no data
2	no data	no data	no data	0	no data
3	no data	no data	no data	0	no data
4	no data	no data	no data	0	no data
5	no data	no data	no data	0	no data
6	no data	no data	no data	0	no data
7	no data	no data	no data	0	no data
8	no data	no data	no data	0	no data
9	no data	no data	no data	0	no data
10	no data	no data	no data	0	no data
11	no data	no data	no data	0	no data
12	no data	no data	no data	0	no data
13	no data	no data	no data	0	no data
14	no data	no data	no data	0	no data
15	no data	no data	no data	0	no data
16	no data	no data	no data	0	no data

Module 32-6: Status DECOM MODULE MAX/MIN Window

Module 32-7: DECOM MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
PROVIDE an overview of the status here			
Base Counter Value		Hex, Octal, Decimal	Listbox choice of hex, octal, or decimal counter value display. Value is displayed in the desired base.
Rollovers		Decimal	Number of times that the counter has rolled over to its' initial value
SeqErrs		Decimal	Number of sequence error that have occurred. A sequence error has occurred if the counter changes in the wrong direction, stays the same in consecutive buffers, or changes by more or less that the defined step size.

Parameter	Unit	Range or Values	Description
PB4 Buffer Time		Time	PB4 time attached to the current buffer. The format is DDD/HH:MM:SS.MMM.mmm where: DDD is the julian day of the year HH is the hour of the day MM is the minute of the hour SS is the second of the minute MMM is the milliseconds of seconds Mmm is the microseconds of milliseconds This time is only displayed. No accommodation is made to calculate the bit offset arrival of any specific sample of any parameter.
Reset Latched Flags		Button	Press this button to acknowledge and clear the latched limits. Clearing is defined as setting the latched limits to the same state as the current limits flags. If a new out-of-limits of a parameter occurs, it will be latched for recognition until this button is pressed again.
MaxMin Display			Activates the MaxMin Display window. This window displays the same parameters in the same bases as the status window. The Initial, Maximum, Minimum, and Average values with status flags are displayed.
MaxMin Init			Clears MaxMin window and begins sampling again.
STATUS INFORMATION LINES 1 TO 16			
Line number		Decimal	Simple index of parameter definition number.
Description		Text	User supplied text describing this parameter.
Value		Value	Value of parameter. Display content varies depending on Base listbox detailed below. If data has not arrived for a parameter, "no data" is displayed.
Units		Text	User supplied units of measure. If the display base is raw, (i.e. decimal, hex, octal, binary) then "raw" is displayed in this field instead of the user defined text for this field.

Parameter	Unit	Range or Values	Description
Flags		Status	Status Flags: "RH" indicates a red high limit violation "YH" indicates a yellow high limit violation "YL" indicates a yellow low limit violation "RL" indicates a red low limit violation "D" indicates a delta limit violation "*" indicates that parameter is saturated "nl" indicates that parameter has no limits enabled "in" indicates the parameter is within red / yellow limits "f" indicates that the value for this parameter has been forced (entered manually in the Config window) rather than an actual decommutated value.
Latched		Status	Displays the flags that were active when the red or yellow violation occurred and the latch tripped.
Base		Decimal, Binary, Descriptor, EU Poly, EU Poly SN, EU Line, EU Line SN, Octal, BCD, Decimal, Signed Dec, Hex, Ascii Text, IEEE FP, IEEE FP SN	Numerical base (or type) of display of the decommutated value.

Module 33 DELAY MODULE

MODULE NAME: Delay.dll

Key Functions

- Delays a data stream by a certain number of milliseconds
- Delays a stream minimally by one queue insertion
- Delays a data stream by a certain fifo queue depth
- Handles different delays types and lengths for four specific streams, allows a differently delay for other, non-specific streams.

Required Hardware

- No hardware required, all data is passed by module connections

Add Module Guidelines

Any number of Delay modules may be used on a desktop. Since one iteration of the Delay module can handle four selected streams, only a few iterations can handle many different streams, up to the limits of system resources.

Required Input Source, Connections, and Events

INPUT SOURCE: All data is sent to the Delay module by module connections.

CONNECTIONS: All modules that are to send data to the Delay module must be connected. The user may choose which output port of the Delay module will contain which streams.

EVENTS: No events are expected or generated.

Module Functional Description

This module is used to simulate delay in networks, satellite roundtrips, or any other test where one or more streams need to be delayed, such as command verification simulation and test.

Three types of delays are possible.

1. A minimal delay capability to slightly delay a buffer of data by one queuing step within the module. This delay varies to provide jitter, based on system loading.

2. A FIFO (first in, first out) delay. In this mode, the user can specify a queue depth, up to a maximum of ten buffers. When data is received, the queue fills. Once the queue is full, the first buffer received is sent, and the newest buffer is loaded on the back of

the queue. This allows a delay, at the data rate, of one to ten buffers. This function is most useful delaying telemetry, or other data that comes at a specific rate.

3. A millisecond time delay. In this mode, the user can specify how late, from 10 milliseconds to 100 seconds, specified in milliseconds, the stream is to be delayed. This function is most useful with command verification simulation and testing, and other uses where buffer come in at a varied rate. The specified millisecond count is rounded to the nearest ten milliseconds, and is subject to system time limitations. Input data processing has a higher priority, so this time period is only approximate.

The user can select the data from up to four other modules. Each module can have its' data delayed differently, eg. 100 milliseconds on one stream, a four deep FIFO on another, a three second delay on a third stream, and an eight deep FIFO on yet another. If data comes in from a module other then those selected by the user, the data will fall into a category called "other", and can have a fifth type of delay assigned to it. The user can elect to leave the time tag on the buffer as is, or it can be set to the current PB4, depending on the simulation required. No buffers are ever copied. The same buffer that was input to this module is output from this module after the delay has been imposed.

Each selected stream can be sent out different ports of the Delay module. The "other" data can also be assigned a port. Only one copy of a stream is sent. The input stream to output port can be one-to-one, but this not necessary. Several of the streams can be sent to one port, allowing multiplexing of the buffers. No buffers are changed in the process, each stream is directed to the chosen port without regard to how the other streams are directed. In this way, different delays and different multiplexing can be accomplished.

All held buffers may be sent, ignoring any remaining delay, by pressing the FLUSH button.

As with any other program which hold buffers for a time period, an overflow situation can occur. Suppose a high rate telemetry stream is input, and a delay of one minute is selected; after the first ten buffers are received for that stream, the eleventh would be dropped and returned to the Avtec PTP for Windows Free Queue. This mechanism protects the rest of the modules and the Avtec PTP for Windows software from just filling up all the buffers and never returning them. Both the FIFO queue for each stream and the Millisecond queue for each stream hold up to ten buffers. A maximum number of delayed buffers can be specified in the range of 10 to 128. This value limits the number of buffers being held by the Delay module. The default maximum buffers is 64. When a buffer arrives, and a queue dedicated to a specific stream has an unused slot to hold the buffer, the maximum number of active buffers is checked; if this value would be exceeded the buffer is counted as an overflow and the buffer is returned to the Avtec PTP for Windows Free Queue for re-use. The user may also specify a minimum time between buffers output, thus guaranteeing buffers cannot be sent back-to-back.

Input Data Format

The Delay module receives buffers of data from various other modules. There are no fields or values expected within the data, only bit-contiguous data. The entire buffer is delayed.

Output Data Format

No changes are made to the data content. The user can optionally paste a new time stamp on the data if desired.

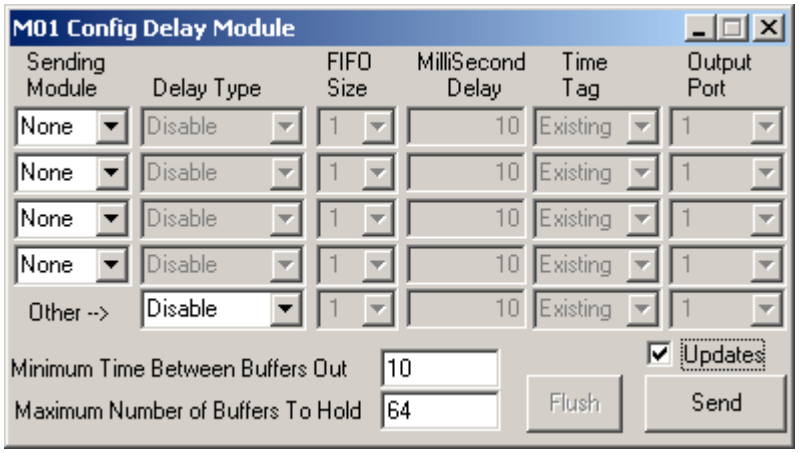
Module Window

When Delay.dll is added to a desktop, the Delay dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 33-1: DELAY MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DELAY MODULE Board Parameters sub-menu.



Module 33-2: Config DELAY MODULE Window

Module 33-3: DELAY MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
GENERAL SETUP			

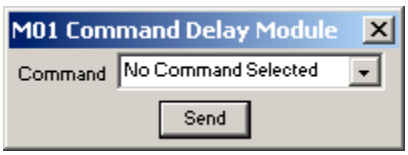
Parameter	Unit	Range or Values	Description
Minimum Time Between Buffer Out	Milliseconds	1 to 500	Specifies the minimum time between buffers sent.
Maximum Number of Buffers to Hold	Decimal	10 to 128; default 64; recommended 64	Specifies the maximum number of buffer that this module can hold, regardless of the type of delay or stream that they come from, or port they are assigned to be output.
Flush	Button	Push	Sends all buffers currently being held, ignoring any remaining delay.
SENDING MODULE SETUP--available options change with selected setup			
Sending Module	Module number	None, or 1 to 32	Selects a module whose input, will be delayed. Up to four specific sending modules may be selected for processing, using different entry rows. Data from a non-selected module is handled on the "other" row.
Delay Type	Selection	Disable, Minimum Delay, FIFO, Millisecond	Disable causes the data from the specific module to be discarded. Can be used for filtering. Minimum Delay causes a small jitter to be imposed by queuing and dequeuing the data. No additional delay is imposed. FIFO permits the module to withhold the last n buffers, where n can be 1 to 10. When a new buffers comes in, the oldest is sent. Millisecond delay waits a certain specified number of milliseconds, subject to system timing limitations and rounded to the nearest 10.
FIFO Size	Decimal	1 to 10	FIFO depth. Specifies the number of buffers to withhold, until more input is received. Applicable only when Data Type is FIFO.
Millisecond Delay	Decimal	10 to 100000	Specifies the approximate number of milliseconds to delay each buffer in this stream, subject to system timing limitations and rounding to the nearest 10. Applicable only when Data Type is Millisecond.
Time Tag	Time	New Existing	Allows the buffer to be sent with the existing time tag, or a new time tag can be generated at the time the buffer is sent.
Output Port	Port	1,2,3,4	Selects the output port for this data stream.

Extended Options Window Definition

There is no extended options window for the DELAY MODULE.

Configuration Notes

When you click the Command button on the DELAY MODULE window, a Command DELAY window appears, as shown below.



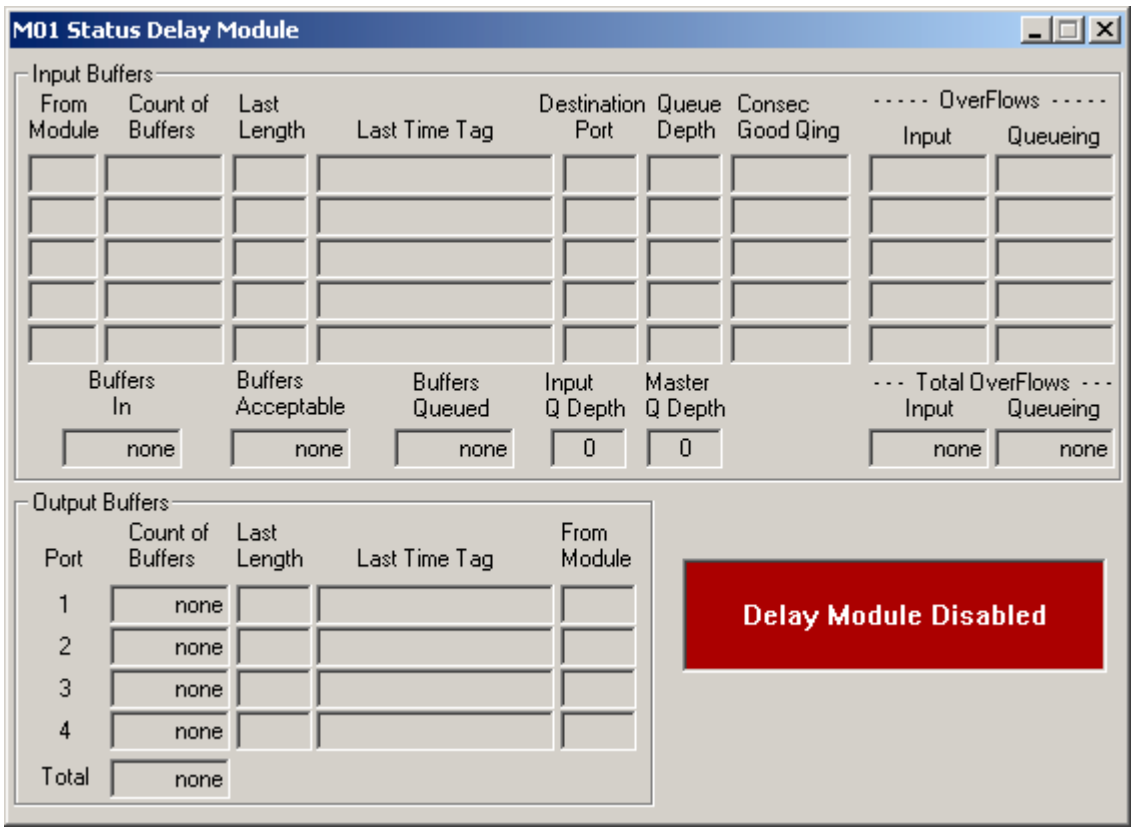
Module 33-4: Commands DELAY MODULE Window

Module 33-5: DELAY MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	N/A	N/A	No operation
Flush All Held Buffers Now	N/A	N/A	Flushes all held buffers within this module. Equivalent to the FLUSH button on Config window.

Status Window Definition

When you click the Status button on the DELAY MODULE window, a Status DELAY window appears, as shown below.



Module 33-6: Status DELAY MODULE Window

Module 33-7: DELAY MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
INPUT BUFFERS			
INPUT BUFFERS/MODULE STATUS			
From Module	Module number	1 to 32	Displays the module number sending data.
Count of Buffers	Decimal		Number of buffers from the module
Last Length	Decimal		Length of last buffer from the module
Last Time Tag	Time		Time Tag of last buffer from the module.
Destination Port	Port		Port number that will be used to send the data. This port must be connected. If the port is not connected, no counts will show, since no data was sent.
Queue Depth	Decimal		Current queue depth. This is how many buffers are backlogged for this stream.
Consec Good Qing	Decimal		Consecutive buffer queued successfully. This is how many buffers were queued without any overflows.
Input OverFlows	Decimal		Count of overflows enqueued onto the Delay input queue.
Queuing OverFlows	Decimal		Count of overflows enqueued onto the specific stream/Fifo/Millisecond queue for delay execution and checking.
INPUT BUFFERS/STATUS SUMMARY			
Buffers In	Decimal		Total count of buffers which came to this module.
Buffers Acceptable	Decimal		Count of Acceptable buffers. Data has to come from a module that is selected (or "other" is selected), has a delay type enabled, and has a valid output port.
Buffers Queued	Decimal		Number of buffers queued into this module.
Input Q Depth	Decimal		Number of buffers on the Delay module input queue.
Master Q Depth	Decimal		Total number of buffers being held by this module in any queue.
Input Total Overflows	Decimal		Number of buffers discarded because a specific queue was full. Usually applicable to millisecond delays.
Queuing Total Overflows	Decimal		Number of buffers discarded because the total number of buffers would be violated.
OUTPUT BUFFERS			
Port	Port		Output Port number
Count of Buffers	Decimal		Count of buffers output on this port.
Last Length	Decimal		Length of last buffer output on this port.

Parameter	Unit	Range or Values	Description
Last Time	Time		Time stamp of last buffer output on this port.
From Module	Module number	1 to 32	Module number from which this buffer originally came
Total	Decimal		Total number of buffers output on this port
Color Status Window	Colored Text		Shows Delay Module Disabled / Enabled with a red or green background

Module 34 DST SFDU FORMATTER MODULE

MODULE NAME: DstSfduFormatter.dll

Key Functions

- Encapsulates input frame with a 100-byte SFDU header
- Automatically fills fixed header fields
- Provides user-configurable header fields
- Automatically fills header fields that change with data input

Required Hardware

- This module does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

Do not exceed the Avtec PTP for Windows limitation of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module may provide frame input to the module. However, typical use is data that has been processed by both the Serial Input and VC Processor modules so that status information is available to fill in the SFDU Header. The module can process input that does not include status information.

CONNECTIONS: Outputs data to port 1, which must be connected to the input of another module or the data is lost.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

The DST SFDU FORMATTER MODULE supports encapsulation of telemetry frames with a Standard Formatted Data Unit header. The header format is based on the definitions as needed for the Deep Space Terminal. Frame synchronization states are recorded within the SFDU header only if a serial input module using the MONARCH-E board is loaded as a source of the telemetry stream. In addition, the Reed Solomon decoder and Extended Status must be enabled.

100 byte header

Note: See *SFOC Software Interface Specification for the Mars Observer Spacecraft Telemetry Data SFDU*, Document SFOC-1-GIF-DSN-MOGCFTlm for a detailed overview of the DST SFDU format.

Input Data Format

Input data are buffers of telemetry, usually synchronous, sometimes asynchronous, sometimes CCSDS.

Output Data Format

Output data are frames of telemetry with DST SFDU header. Refer to *SFOC Software Interface Specification for the Mars Observer Spacecraft Telemetry Data SFDU*, Document SFOC-1-GIF-DSN-MOGCFTlm for more details.

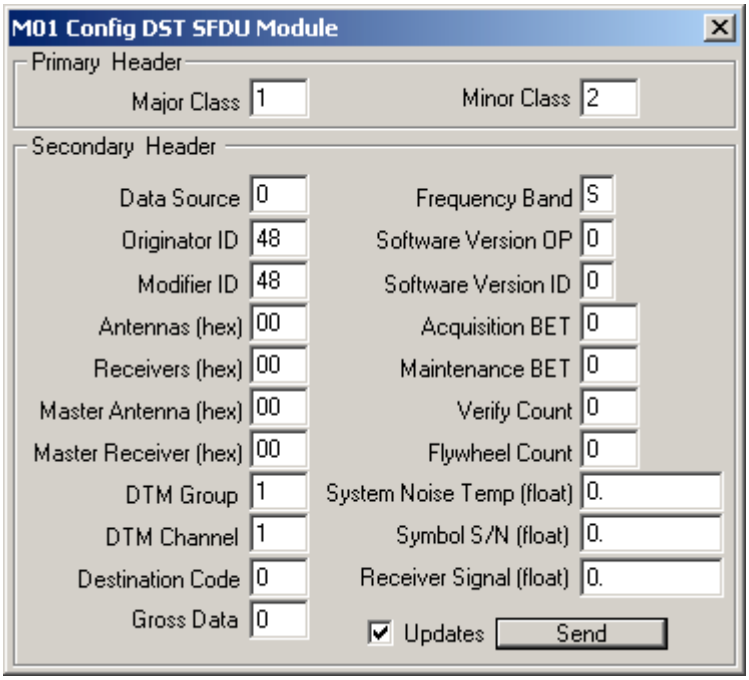
Module Window

When *DstSfduFormatter.dll* is added to a desktop, the DST SFDU Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 34-1: DST SFDU FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DST SFDU FORMATTER MODULE Board Parameters sub-menu.



Module 34-2: Config DST SFDU FORMATTER MODULE Window

Module 34-3: DST SFDU FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
All configuration parameters define fields of the SFDU. The config window parameter name maps to a SFDU field name. Refer to the DST SFDU documentation for details and limitations of each field. The values entered in this window are stamped into each SDFU output from the module.			
PRIMARY HEADER			
Major Class	Decimal	8-bit field Byte 29	Major Data Class; Default = 1 (Spacecraft telemetry data)
Minor Class	Decimal	8-bit field Byte 30	Minor Data Class. Indicates configuration of DSCC (Deep Space Communications Complex) frame sync and Reed-Solomon decoding processes. Valid values = 0, 1, 2
SECONDARY HEADER			
Data Source	Decimal	8-bit field Byte 40	Data Source number
Originator ID	Decimal	8-bit field Byte 37	Indicates message origin; 48=originated from DSN (Deep Space Network)
Modifier ID	Decimal	8-bit field Byte 38	Indicates where message was last modified; 48=last modified by DSN
Antennas	Hex	8-bit field Byte 85	Antennas in use
Receivers	Hex	8-bit field Byte 86	Receivers in use

Parameter	Unit	Range or Values	Description
Master Antenna	Hex	8-bit field Byte 87	Indicates which DSS antenna data was received from
Master Receiver	Hex	8-bit field Byte 88	Indicates which receiver was used with Master Antenna
DTM Group	Decimal	8-bit field Byte 89	DSCC Telemetry Subsystem Group number
DTM Channel	Decimal	8-bit field Byte 90	DTM Channel number
Destination code	Decimal	8-bit field Byte 71	Destination Code
Gross Data	Decimal	8-bit field Byte 72	Gross data
Frequency Band	ascii	8-bit field Byte 66	Indicates frequency band of data; values = S, X, K
Software Version OP	ascii	8-bit field Byte 93	Software Version Operational Level; assigned by DSN Operations
Software Version ID	ascii	8-bit field Byte 94	Software Version Identifier; assigned by DTM CDE
Acquisition BET	Decimal	0 to 15 Byte 55	Acquisition Bit Error Tolerance – defines number of allowable bit errors in sync pattern in search and verify modes; values=0 – 15
Maintenance BET	Decimal	0 to 15 Byte 56	Defines number of allowable bit errors in sync pattern in lock and flywheel modes; values=0 – 15
Verify Count	Decimal	0 to 15 Byte 57	Number of frames required for frame sync logic to transition from verify to lock state; values=0 – 15
Flywheel Count	Decimal	0 to 15 Byte 58	Number of frames required for frame sync logic to transition from flywheel to search state; values=0 – 15
System Noise Temp	Float kelvin	32-bit field Bytes 73-76	System Noise Temperature in Kelvin
Symbol S/N	Float db	32-bit field Byte 77-80	Estimated signal-to-noise ratio (dB) in symbol domain (prior to Viterbi decoding)
Receiver Signal	Float dBm	32-bit field Byte 81-84	Receiver signal level (dBm) for master receiver in array

Extended Options Window Definition

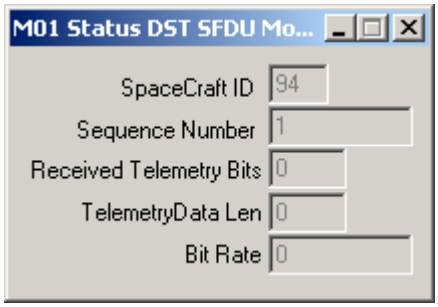
There is no extended options window for the DST SFDU FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the DST SFDU FORMATTER MODULE.

Status Window Definition

When you click the Status button on the DST SFDU FORMATTER MODULE window, a Status DST SFDU FORMATTER window appears, as shown below.



Module 34-4: Status DST SFDU FORMATTER MODULE Window

Module 34-5: DST SFDU FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Spacecraft ID	Decimal		Displays Spacecraft ID
Sequence Number	Decimal		Displays sequence number
Received Telemetry Bits	Decimal		Count of received bits
Telemetry Data Len	Decimal		Displays telemetry data length
Bit Rate	Decimal		Bit Rate in bits per second

Module 35 DUMP TO LINE PRINTER MODULE

MODULE NAME: DumpToLP.dll

Key Functions

- Dumps input buffer in hexadecimal, octal, or decimal
- Output can be to a line printer, a disk file in text format, or any network path

Required Hardware

- A line printer, if desired. If no printer is present, the output may be directed to any disk file, or network path.

Add Module Guidelines

No quantity restriction on module iterations. One module will dump all input to one file or device. The module that sent the data is identified in the output.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives buffers of data from other modules.

CONNECTIONS: One or more modules must be connected to the input of this module. No output connections are required; any output connections are ignored. No buffers are output from this module.

EVENTS: No events are expected or generated.

Module Functional Description

The Dump to Line Printer (DumpToLP) Module accepts blocks, frames, commands, network traffic, or any other inter-module data and outputs a formatted listing to a line printer (default) or to a file in octal, hexadecimal, or decimal format. User specified fields include number of bytes per line and number of lines per page. Current PB4 buffer attached time, and the module number and name from which the data came are listed with each frame/block/command. There is also an option to immediately print (character-at-a-time printer) and to spool full pages (laser printers).

Care should be exercised when printing to a slow device, such as a printer. To prevent this module from becoming deluged with buffers, and to protect real time data handling modules from running out of buffers, a copy of the input buffer is made. The buffer is immediately released. Data buffers coming in while the print is active are dropped without being printed. Samples of the data will be printed at the rate that the output device can handle the data. If data could be sent to this module faster than the attached printer is capable of printing the

buffer content, and the user wants to dump all the data, it is suggested that the output be directed to a file.

Input Data Format

No specific format is expected. Any Avtec PTP for Windows buffer from any other module is acceptable. The module makes no attempt to identify the type of data or analyze the content of any part of the buffer.

Output Data Format

The actual output is to any device or path acceptable to Window NT. To select the LPT1 parallel port, the Window name is \\. \\LPT1. This is the default output device. It may be changed to any other file on any disk or another computer on the network.

Sample of actual output of two frames:

```
045:19:29:13.866 -- 9 bytes Module 1 [C:\Ptp_nt\Modules\M0188_ManualCommand.dll]
FA F3 20 02 11 43 9C EB 90

045:19:29:39.654 -- 8 bytes Module 1 [C:\Ptp_nt\Modules\M0188_ManualCommand.dll]
23 51 87 BB AC D4 8E F2
```

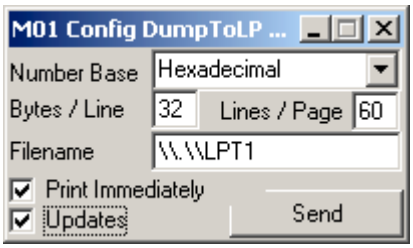
Module Window

When DumpToLP.dll is added to a desktop, the Dump to LP dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 35-1: DUMP TO LP MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure DUMP TO LP MODULE Board Parameters sub-menu.



Module 35-2: Config DUMP TO LP MODULE Window

Module 35-3: DUMP TO LP MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Number Base	User Selection	Hexadecimal Octal Decimal	Format to output data: Hexadecimal, Decimal, or Octal
Bytes/Line	Decimal	10 to 250	Number of bytes to output per line (defines page width)
Lines/Page	Decimal	Not limited	Number of lines per page (defines page length)
Filename		Valid NT path	Output location: Default=LPT1 (parallel port for printer); user can input any filename with path to output data
Print Immediately		Enable Disable	If enabled, each line is output one at a time; suitable for a character printer or output to a file. If disabled, output is held until a full page is available, then the entire page is output; suitable for a laser printer.

Extended Options Window Definition

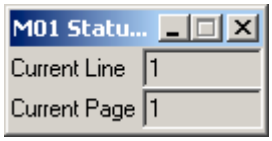
There is no extended options window for the DUMP TO LP MODULE.

Configuration Notes

There are no configuration notes for the DUMP TO LP MODULE.

Status Window Definition

When you click the Status button on the DUMP TO LP MODULE window, a Status DUMP TO LP window appears, as shown below.



Module 35-4: Status DUMP TO LP MODULE Window

Module 35-5: DUMP TO LP MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Current Line	Decimal	Begins at 1	Current line number being output
Current Page	Decimal	Begins at 1	Current page number being output

Module 36 ERROR INJECT MODULE

MODULE NAME: ErrorInject.dll

Key Functions

- Accepts module buffer data input and either throughputs the data or outputs the data with errors injected per operator definition
- Provides error insertion at the time of operator manual command or optionally provides automatic implementation of errors by frame number. Automatic errors are defined by a script file that must be created prior to data flow.
- Outputs only a pre-determined number of data buffers (optionally)
- Inverts the data in a defined number of input buffers (optionally)
- Deletes an operator defined number of input buffers prior to output (optionally)
- Corrupts (changes bytes) an operator defined number of input buffers (frames) prior to output (optionally)
- Provides Additive White Gaussian Noise insertion (optionally)
- Shifts a defined number of data buffers by a defined number of bits prior to output (optionally)

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the 32 modules per desktop limitation: there are no other limitations to adding ERROR INJECT MODULES to a desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from any other module.

CONNECTIONS: Output is to port one.

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The ERROR INJECT MODULE is used to induce errors in a data stream. The ERROR INJECT MODULE is connected within the data path, i.e. it takes the data stream in on its input port, and transmits the data stream on its output port. This module receives data from any other module, implements the defined errors on the data, and then outputs the same data that is changed only per error definitions. The data stream passes through the module untouched unless the operator defines and enables error insertions.

The errors that can be induced in the data stream are set up in the configuration window. The available errors follow:

1) Discards and does not output a defined number of input buffers (Skip)

2) Accepts and outputs a defined number of input buffers, and then does not output any data after that number (Accept)

3) Inverts a defined number of input buffers prior to output (Invert)

4) Corrupts data bits for a defined number of input buffers. Corruption is defined by bit offset from buffer start, number of bits to corrupt, and a value that is XOR'd onto the data to create the corruption (Corrupt)

5) Overwrites a fixed pattern into a fixed location of each data buffer for a defined number of buffers (Replace)

6) Shifts a defined number of input buffers by a fixed number of bits. The shift is always to the right (bits inserted on the first data buffer).

7) Generates Additive White Gaussian Noise

As listed above, the ERROR INJECT MODULE includes the option to inject Additive White Gaussian Noise (AWGN) on the input data stream. The AWGN option simulates sending a data stream through a Binary Symmetric Channel (BSC) with AWGN by statistically perturbing the data to produce errors corresponding to the given Es/No, which is the energy per symbol to No (Noise Density, i.e. Noise Power In a 1 Hz Bandwidth) ratio in dB.

The module provides control of error insertion via two different methods. The first is the typical control that is available through the configuration window. For this method, the operator defines the error and then enables the error (depress the error button) and the error is injected at the moment that it is enabled. This means that error insertion does not start when module is enabled, rather the error insertion starts when the error type is enabled from the configuration window.

The second method for defining error injection is to use a script file, which provides the operator more control. For this method, the operator creates a script file and saves it to a known directory. The script file allows the operator to implement the same errors as the configuration window and uses the same parameters to define the errors. However, the script file also provides a start frame number, which gives the operator control over the exact start and end of each error injection relative to the start of data input into the module. The script file name and location is entered in the configuration window and set up in an enabled state. For this case, as soon as the module is enabled, the script file is implemented. Refer to the ERROR INJECT MODULE configuration notes section for details of the script file contents.

Note that the script file can be configured to be started as soon as the module is enabled, but it also allows the operator to keep the script disabled at module enable, and then the script is

not implemented until the operator enables it from the configuration window. If the script file is not enabled at module enable, then the start frame numbers are no longer relative to the first data buffer input to the module; in this case, it is relative to the first frame input to the module after the script file is enabled. The default setup in the configuration window is to enable script file at module enable. However, the operator should still verify that the script file is configured as expected prior to starting a data flow.

Input Data Format

Any data buffers (from another module) can be input.

Output Data Format

Output is the input data throughput with errors inserted. Errors are injected in the output data according to operator definition. If no errors are defined, then data is throughput without change.

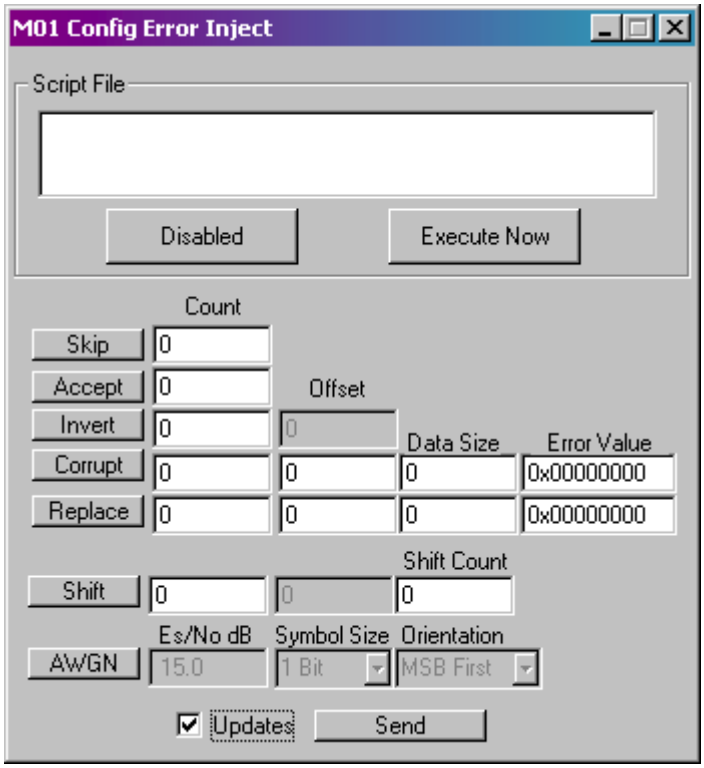
Module Window

When *ErrorInject.dll* is added to a desktop, the Error Inject dialog appears. Left mouse click the command button to access the error injection options. This module does not have a configuration or status window.

Module 36-1: ERROR INJECT MODULE Window

Configure Window Definition

The ERROR INJECT MODULE Configuration window is shown below.



Module 36-2: Configure ERROR INJECT MODULE Window

Module 36-3: ERROR INJECT MODULE Configuration Window Parameters

Parameter	Unit	Range or Values	Description
Script File	Text File Name; name includes path	Files that have previously been created and have a known directory location and name	If not disabled, then this script file starts on buffer one as soon as the module is enabled. Text name of the script. Refer to "Configuration Notes" section for definition of script file contents. Script file runs on start and lets the user define specifically which frame has errors relative to the first buffer Disabled: When this button is selected, it means the script file is NOT enabled as soon as the module is enabled. The script file does not start to run until "Execute Now" is enabled. Execute Now: When selected, this button causes the script file to implement. This can be used to start the script file when the module is enabled with "Disabled" on. It can also be used to run the script file again if it has already executed and has reached the end.
NOTE: Issuing a Send (depressing the "Send" button) does not insert the error. For all error inject options, the error is ***not*** implemented until the operator enables that error type by depressing the button labeled with that error type (left-most buttons in the window). For example, if 10 is entered in the Skip option, the error is inserted when the "Skip" button is depressed. Once an error is enabled, the counts entered in the setup fields for that error all revert to zero.			

Parameter	Unit	Range or Values	Description
Skip	Input buffers	User Defined	Throws away buffers input to the module. Count: The number of consecutive input buffers that are discarded and not output.
Accept	Input buffers	User Defined	Accepts a user-configured number of input buffers and output those buffers, and then stops accepting input When this command is issued, the module stops data output after the COUNT number of input buffers. To restart the data output, issue a re-enable. Count: The number of consecutive input buffers to accept and output. The first buffer input is input buffer #1.
Invert	Input buffers	User Defined	Inverts all bits of data in the data buffer. Count: Defines the number of consecutive buffers to apply the invert option. Once the Invert button is depressed. The "Count" number of input data buffers is inverted.
Corrupt	Input buffers	User Defined	XORs a bit pattern onto the buffers at a fixed bit location to create data corruption Count: The number of consecutive input buffers that are corrupted. Offset: number of bits from the beginning of the buffer Data Size: how many consecutive bits to corrupt Error Value 0x00000000: Each 0 represents 4 bits. This is the value that is XOR'd with the number of bits to be corrupted. It uses the right-most bits. For example, if three bits to be corrupted, then the right most 3 bits are XOR'd
Replace	Input buffers	User Defined	Overwrites a fixed pattern onto a fixed number of bits at a fixed location in each data buffer Count: The number of consecutive input buffers to which this corruption is applied. Offset: number of bits offset from start of buffer to the corruption Data Size: number of bits to corrupt Error Value 0x00000000: Forces the bits to this specified value.
Shift	Input buffers	User Defined	Shifts data buffers to the right by a fixed number of bits Count: The number of consecutive input buffers that are shifted Shift Count: defines the number of bits to shift the buffer. Value from 1 to 32. Creates a long frame for first frame: inserts bits at the start of the buffer. Always does a shift right. Number of bytes out is always the same. First buffer has X number of bits inserted and then seven bits are carried over to the next buffer. At end of corruption, there will be a short frame because that buffer truncates the extra bits.

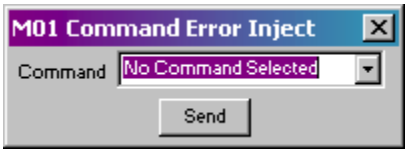
Parameter	Unit	Range or Values	Description
AWGN	Each field of AWGN has a different unit. Refer to definitions	Each field has a different range. Refer to definitions	***NOTE:* This option requires a Send**. This option implements differently than the other options. To turn on the option configuration, depress the AWGN button. Once the button is selected, enter the setup fields, and then to enable the option, issue a Send. Es/No dB: the desired energy per symbol to noise density in dB Symbol Size: 1 Bit to 8 Bit 1 Bit Mode outputs the data just as it came in with errors injected into each byte as necessary. 2 to 8 Bit Mode outputs symbols with 1 bit of input corresponding to a byte symbol output. It produces 2-8 bit soft symbols from the given data. Orientation: this tells the order in which the symbols should be output from the input data stream in 2-8 Bit Symbol Size Mode. MSB First - means unpack the input byte from MSB-LSB LSB First - means unpack the input byte from LSB-MSB

Extended Options Window Definition

There is no extended options window for the ERROR INJECT MODULE.

Configuration Notes

The figure below shows the ERROR INJECT MODULE command window. Currently, the command window has no options to implement.



Module 36-4: Command ERROR INJECT MODULE Window

The ERROR INJECT MODULE may use a script file to define error types and location within the data stream. An example script file is provided and the example provides definitions of the different parameters (keywords) that can be included in a script. The script is created "offline" from the Avtec PTP for Windows and can be generated using a notepad type of system application program. To use a script file, the operator must know the file's exact name and directory location, and this information must be typed exactly into the configuration window.

All script file counts are relative 0, which means that if Frame 10 is defined as the start frame, the start frame is buffer 11 of the input stream.

When generating the script file, make sure to enter a return at the end of every line. If a return is not entered, then that line will not be implemented when the script is run.

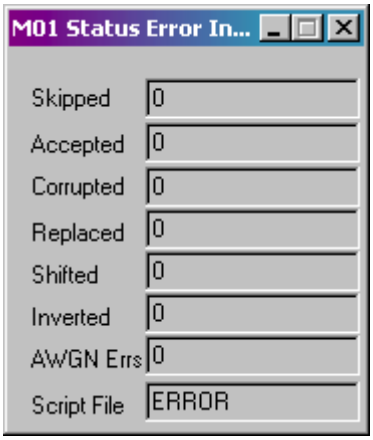
For all script file keywords, make sure to use capitals for the keyword names. Keywords without capitals are not recognized and are not implemented.

EXAMPLE Script File Format and Keyword Definitions

```
# This is an example script file for M0003_ErrorInject
# Per frame errors are inserted in this order –
# Skips Accepts Corrupts Replaces Shifts Inverts

# Any keyword not entered is presumed to be 0
# Keywords may occur in any order but must be on the same line

# Count is the number of consecutive frames with an error
# Offset is the number of bits relative to offset from the start of frame
# Size is the number of consecutive bits to include in a pattern
# Pattern must be in hex
# ShiftCount is in bits

# Frame is a relative input counter.
# Frame is circular and rolls over from 255 to 0.

###### Example of all keywords ##############
#Frame=5
#Skip Count=50
#Frame=10
#Replace Count=20 Offset=40 Size=7 Pattern=0x12345678
#Frame=20
#Corrupt Count=11 Offset=41 Size=8 Pattern=0xfedcba9
#Accept Count=102
#Shift Count=12 Offset=42 ShiftCount=6 #Frame=30
#Invert Count=12
##############################################

Frame=5
Skip Count=50

Frame=10
Replace Count=20 Offset=40 Size=7 Pattern=0x12345678

Frame=20
Corrupt Count=11 Offset=41 Size=8 Pattern=0xfedcba9
Accept Count=102
Shift Count=12 Offset=42 ShiftCount=6

Frame=30
Invert Count=12
```

Status Window Definition

The ERROR INJECT MODULE status window is shown below.



Module 36-5: Status ERROR INJECT MODULE Window

Module 36-6: ERROR INJECT MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Skipped	buffers	Incrementing counter	Total number of input buffers that were discarded and not output
Accepted	buffers	Incrementing counter	Total number of input buffers (frames)
Corrupted	buffers	Incrementing counter	Total number of input buffers that were corrupted per the "corrupt" definition in the configuration window prior to output
Replaced	buffers	Incrementing counter	Total number of input buffers that were replaced per the "replace" definition in the configuration window prior to output
Shifted	buffers	Incrementing counter	Total number of input buffers that were shifted per the "shift" definition in the configuration window prior to output
Inverted	buffers	Incrementing counter	Total number of inverted buffers output. Number of inverted buffers relative to the last zero. Note that an enable performs a zero
AWGN Errs	Errors	Incrementing counter	The number of AWGN Errors introduced into the data stream by the module.
Script File	Text filename	Filename entered in configuration window	Text file name if a file is present. ERROR if it can't open the script file, but this does not mean that there is an error in data

Module 37 EDOS SERVICE HEADER (ESH) FORMATTER MODULE

MODULE NAME: ESHFormatter.dll

Key Functions

- Prepends EDOS-defined, 20-byte ESH to each data unit
- Accepts packet or frame input and outputs data unit with ESH appended
- Identifies partial packets and location of fill in ESH
- Completes ESH data fields using data processing results
- Provides user-configurable ESH data fields that indicate test data, playback data, or recovery processing

Required Hardware

- No special hardware is required.

Add Module Guidelines

A desktop supports a maximum of 32 modules. No other limitations apply.

Required Input Source, Connections, and Events

INPUT SOURCE: Packet input is typically input from either the Packet Processor Module or the Packet Processor Version 2 module. Frames or CLCWs can also be input from a module.

CONNECTIONS: Output is to port one.

EVENTS: This module neither requires an event input nor generates an event output.

Module Functional Description

The EDOS SERVICE HEADER (ESH) FORMATTER MODULE is a packet or frame service header module. This module accepts packets from the Packet Processor Module or from formatted packet files, VCDUs or CLCWs from the Scatter Gather Module. The module generates a header for each data unit. The header contains data statistics, including number of VC sequence errors, and number of packet sequence errors. The output of this module is the original data unit input with the EDOS Service Header (ESH) prepended.

For packet input, the ESH Formatter Module presumes packet input has previously been processed by the Packet Processor Module (or Packet Processor Version 2 Module) and the VCP Module (or VCP Version 2 Module). To complete the header, status from both of these modules is required. For packet input, the module also checks the packet sequence count and report sequence errors in the ESH

Note: See Appendix B for a detailed overview of the EDOS Service Header.

Input Data Format

Data input is either packets, frames, or CLCWs.

Output Data Format

The input data unit is output with a 20-byte header prepended.

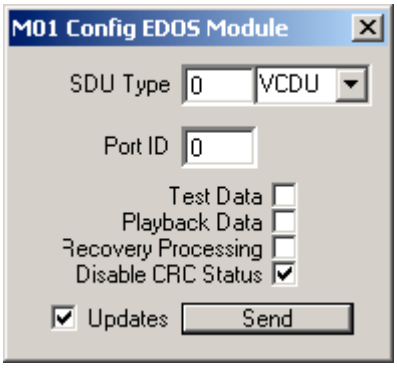
Module Window

When ESHFormatter.dll is added to a desktop, the ESH Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 37-1: ESH FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure ESH FORMATTER MODULE Board Parameters sub-menu.



Module 37-2: Config ESH FORMATTER MODULE Window

Module 37-3: ESH FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
SDU Type	Numeric entry, or Selectable Setup	VCDU, CCSDS packet, CLCW, Other (4 bits)	Defines SDU (Service Data Unit) type, i.e. type of data being received by this module – 0 = VCDU, 1 = CCSDS packet, 2 = CLCW
Port ID	Numeric entry	8 bits	EDOS physical port identification; Range 0 – 63
Test Data	Selectable Setup	Enable, Disable	When checked, the "Test Data" bit is set (1) in each ESH appended to data unit. When unchecked, the header bit is never checked (0). Standard definition is: 0 = live operational data, 1 = EDOS test data.
Playback Data	Selectable Setup	Enable, Disable	When checked, the "Playback Data" bit is set (1) in each ESH appended to data unit. When unchecked, the header bit is never checked (0). Standard definition of the Playback Data bit is: 0 = non-playback data; 1 = playback data/data from on-board spacecraft recorder**.**
Recovery Processing	Selectable Setup	Enable, Disable	When checked, the "Recovery Processing" bit is set (1) in each ESH appended to data unit. When unchecked, the header bit is never checked (0). Standard definition of the Recovery Processing bit is: 0 = live data, 1 = data capture playback data.
Disable CRC Status	Selectable Setup	Enable, Disable	When checked, the module does not perform VCDU CRC status reporting. If this box is checked, then the CRC error flag is never set in the ESH. When unchecked, the module automatically pulls the CRC status from the VCP Module status (which indicates if there was a CRC error in the VCDU). If there was a VCDU error, then the module sets the ESH CRC failure flag to true (1).

Extended Options Window Definition

There is no extended options window for the ESH FORMATTER MODULE.

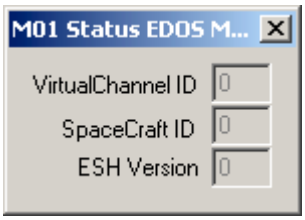
Configuration Notes

A non-zero value (entered by the ESH FORMATTER MODULE) in the "offset to fill" ESH field of an output packet indicates that the output packet is a partial packet. The "offset to fill" field is set to the number of bytes that are real data. The byte location points to the end of real data and to the start of fill data. The byte count does not include the ESH: it starts at byte one of the packet (header is stripped).

If packet input is from Packet Processor Version 2 module, the "delete partial packets" must not be set or partial packets are never input to the ESH FORMATTER MODULE.

Status Window Definition

When you click the Status button on the ESH FORMATTER MODULE window, a Status ESH FORMATTER window appears, as shown below.



Module 37-4: Status ESH FORMATTER MODULE Window

Module 37-5: ESH FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Virtual Channel ID	VCID decimal	6 bits	Displays the VCID number The VCID value is read from VCP Module buffer descriptor (status), which means data must have passed through VCP Module prior to input to ESH Formatter.
Spacecraft ID	SCID decimal	8 bits	Displays the Spacecraft ID number The SCID value is read from VCP Module buffer descriptor (status), which means data must have passed through VCP Module prior to input to ESH Formatter.
ESH Version	Version decimal	4 bits	Currently, status value is always zero. This field provides the EDOS Service Header version number (0).

Module 38 FILE PLAYBACK MODULE

MODULE NAME: FilePlayBack.dll

Key Functions

- Reads data back from a file
- Filters based on the source module number read from MUX header
- Optionally reads the Avtec MUX header and outputs the size defined in the header and/or strips the MUX header from file data
- Provides several read-back modes that include read one time, continuous loop, and read with start and stop bytes

Required Hardware

- The FILE PLAYBACK MODULE does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of FILE PLAYBACK MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: An existing file. For telemetry applications, it is expected that the file contains telemetry data.

CONNECTIONS: What module connections must be made?

EVENTS: Requires an event to drive output.

Module Functional Description

The FILE PLAYBACK MODULE supports reading binary data from hard disk or CD-ROM and transferring it to another module. The FILE PLAYBACK MODULE supports five operating modes. In Fixed Mode, the Playback Module reads data until the end of file and then stops. In Loop Mode, the Playback Module reads until the end of file and then seeks back to the start of the file until disabled. In Segment Mode, the Playback module reads a segment of a file starting at a user-defined start location until a user-defined stop location. Sequence Mode is a combination of Segment and Loop Mode – the selected segment of a file is played in a continuous loop until disabled. Spooled mode allows the FILE PLAYBACK MODULE to be used in conjunction with the File Recorder to transmit frame data from a file while frames are still being written to the file. For this mode, the FILE PLAYBACK MODULE must be loaded after the File Recorder Module on the Desktop, with identical file names. As frames are requested

from the File Playback, it will pull them from the binary file as the File Recorder appends new frames.

The FILE PLAYBACK MODULE requires an event input. It reads a buffer from the file on each event. The File Playback has a single data output, the data read from the file.

Input Data Format

Data source is a file.

Output Data Format

Output depends on the data stored to file:

1. If data buffers (data units) were stored to file with no MUX header attached to each, then this module outputs the event buffer size.

2. If data buffers were stored with the MUX header attached, then this module outputs the size defined in the MUX header.

Module can be set up to strip MUX header prior to output.

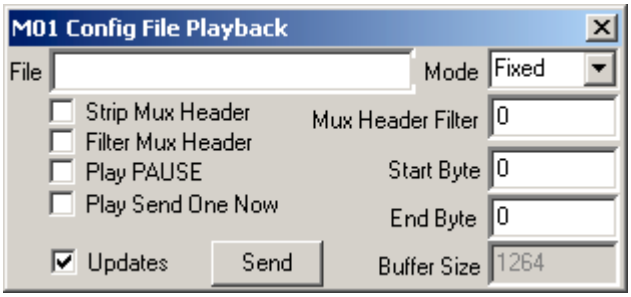
Module Window

When *FilePlayback.dll* is added to a desktop, the File Playback dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 38-1: FILE PLAYBACK MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure FILE PLAYBACK MODULE Board Parameters sub-menu.



Module 38-2: Config FILE PLAYBACK MODULE Window

Module 38-3: FILE PLAYBACK MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
File	text	Existing path and filename	Provides the logical disk, directory path, and file name of the file that is to be opened, read, and output from module
Mode	Selectable setup	Fixed Segment Loop Sequence Spooled	Defines the play back mode, i.e., how the file is read. 1) Fixed – starts at beginning of file, and plays to the end 2) Segment – Using StartByte and EndByte, the portion between is played 3) Loop – Plays entire file over and over again, until manually terminated 4) Sequence – Plays from StartByte to StopByte over and over again until manually terminated 5) Spooled – Used in conjunction with File Recorder, frame data can be spooled from a file that is currently open by the File Recorder for writing
MUX Header Filter			Applies to files that were written from multiple module sources. Each data buffer will have been written with the MUX header and the header includes the source module number. If enabled, only outputs the data from the module source identified in "Filter MUX Header" setup field. See the field definition below.
Start Byte			Start byte in file (used when mode is Segment or Sequence)
End Byte			End byte in file (used when mode is Segment or Sequence)
Buffer Size			Can not be configured.
Strip MUX Header			If enabled, looks to see if MUX header is present and strips it if it is. If disabled, then the MUX header is transmitted with the data.
Filter MUX Header			Used in conjunction with MUX Header Filter field. Identifies the module number to be read back from files that recorded data from multiple module sources.
Play Pause			Stops playback, leaving file pointer in position. Stops data during the flow. If enable desktop with this configuration parameter set, then data flow does not start until the check is removed.
Play Send One Now			Sends one frame at a time (only works in PAUSE mode)

Extended Options Window Definition

There is no extended options window for the FILE PLAYBACK MODULE.

Configuration Notes

There are no configuration notes for the FILE PLAYBACK MODULE.

Status Window Definition

When you click the Status button on the FILE PLAYBACK MODULE window, a Status FILE PLAYBACK window appears, as shown below.

Module 38-4: Status FILE PLAYBACK MODULE Window

Module 38-5: FILE PLAYBACK MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
File			Logical disk, path, and file name that the module is reading (or is trying to read)
Frame size			Transmitted size. If MUX header present, is data buffer plus MUX header. If no MUX header present, then it is the data buffer size.
Byte Position			Displays the current read (pointer) location in file
File Size			Displays total byte count of the file to be read back
Loop			Displays number of times file has been played when in Loop or Sequence mode
Play			Displays mode description for Fixed, Segment, Loop, Sequence, or Spooled modes
State			Displays "Waiting", "Running", "Paused", or "Done"
Progress bar			Dots that indicate relative position in the file
Empty Status bar Window			Lists the module numbers read from the MUX headers saved to file

Module 39 FILE PLAYBACK VERSION 2 MODULE

MODULE NAME: FilePlayBackVer2.dll

Key Functions

- Reads data back from a file
- Supports file sizes >4 GB
- Optimized for dual processor systems (PTP-EX)

Required Hardware

- The FILE PLAYBACK VERSION 2 MODULE does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of FILE PLAYBACK VERSION 2 MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: An existing file. For telemetry applications, it is expected that the file contain telemetry data.

CONNECTIONS: The FILE PLAYBACK VERSION 2 MODULE can send its output to any other module.

EVENTS: Requires an event from another module to drive output. Does NOT generate an event output.

Module Functional Description

The FILE PLAYBACK VERSION 2 MODULE supports reading binary data from hard disk and transferring it to another module. The FILE PLAYBACK VERSION 2 MODULE is an extension of the original File Playback Module. The Version 2 enhancements include the ability to support file sizes greater than 4 GB and improved performance on dual processor machines.

The FILE PLAYBACK VERSION 2 MODULE supports various operating modes. In Fixed Mode, the Playback Module reads data until the end of file and then stops. In Loop Mode, the Playback Module reads until the end of file and then seeks back to the start of the file until disabled. In Segment Mode, the Playback Module reads a segment of a file starting at a user-defined start location until a user-defined stop location. Sequence Mode is a combination of Segment and Loop Mode – the selected segment of a file is played in a continuous loop until disabled.

The FILE PLAYBACK VERSION 2 MODULE requires an event input. It outputs a buffer of reads a buffer from the file on each event. The File Playback has a single data output, the data read from the file.

Input Data Format

Data source is a file.

Output Data Format

The output data format changes with module configuration and with the source of the original file. Data output from this module is unaltered from the file being read unless the operator has setup one of the stripping options available through module configuration. If the module is configured to read the MUX header, it uses the length defined in the header to determine the size of the buffer to output. Otherwise, data is output in the buffer size that is defined by the event input. Possible output formats are as follows:

1) Data buffers with either MUX Header or MUX Header Ver2 pre-pended to each
2) Data buffers without a MUX Header
3) Ex Log File buffers that include hardware status that is appended at the end of every frame
4) Ex Log File buffers without status appended

For best accuracy of MUX Header descriptions, the operator should refer to the AV_PTP.h header file that is located in ptpapi directory of the Avtec PTP for Windows software. An overview of the headers is also provided in the File Recorder Ver 2 module description.

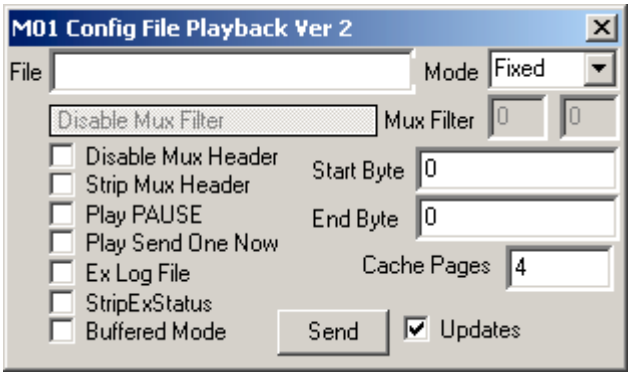
Module Window

When *FilePlayBackVer2.dll* is added to a desktop, the File Playback Version 2 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 39-1: FILE PLAYBACK VERSION 2 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure FILE PLAYBACK VERSION 2 MODULE Board Parameters sub-menu.



Module 39-2: Config FILE PLAYBACK VERSION 2 MODULE Window

Module 39-3: FILE PLAYBACK VERSION 2 MODULE Configure Window Parameters

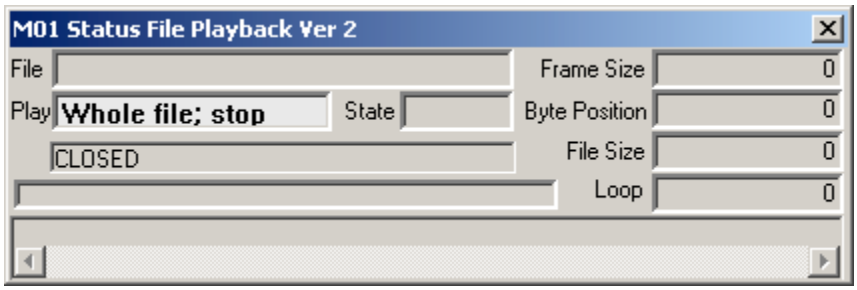
Parameter	Unit	Range or Values	Description
File	Text	Existing path and filename	Provides the logical disk, directory path, and file name of the file that is to be open, read, and output from module
Mode	Selectable setup	Fixed Segment Loop Sequence Burst	Defines the play back mode, i.e., how the file is read. 1) Fixed – starts at beginning of file, and plays to the end 2) Segment – Using StartByte and EndByte, the portion between is played 3) Loop – Plays entire file over and over again, until manually terminated 4) Sequence – Plays from StartByte to StopByte over and over again until manually terminated 5) Burst – Insures complete file is played back in cases where output buffer size is not an integer divisor of file size. This means that data is output in the event (buffer) size defined until there is an amount of data left in the file that is less than the event size, and then only that amount of data is output.

Parameter	Unit	Range or Values	Description
Disable MUX Filter **Filter Modules and Ports** **Filter Modules Only**	Three-option setup button	Disable MUX Filter, Filter Modules and Ports, Filter Modules Only	If no MUX headers present than this setup is invalid. If MUX header present, then these options can be used to filter the data file data buffers. If enabled, the module only outputs the data from the module source identified in "MUX Header Filter." Header version one only has module source ID info and limits which options can be selected. The button setup is changed by selecting it. It is a three-phase setup field and it cycles to the next setup option each time it is selected. The three setup options define how and if any playback filtering is enabled. Filtering options apply only to files that were written with the MUX header appended to each buffer. Disable MUX Filter: This option performs no filtering and should be selected any time MUX headers are present and no data filtering desired. Filter Modules and Ports: Applies to MUX Header Version 2 only. Allows the operator to define both the source module and the output port on that module that is read back from the file. All other data, which includes data from that same module, but a different output port, is filtered and not output. Filter Modules Only: The number entered is the source module number that is read back, and then all other module data is filtered and not output. NOTE: The value entered here is the desktop module number minus one (zero based). For example, if the desired data source is module number one, then enter a zero in this field.
MUX Filter	Integer number for both setup windows	0 to the number of modules in desktop 0 to the number of output ports on the source module	The "MUX Filter" setup option is directly tied to the three-option "Filter" button that is defined above. The available setup fields for "MUX Filter" changes depending on the three-stage Filter button selection. For "Disable MUX Filter," this option is completely disabled and both of the setup fields are grayed out. For "Filter Modules Only," the left-most window is available for setup. For "Filter Modules and Ports," both setup windows are available. Two setup windows are provided for the MUX Filter Option. The left-most window sets up the Module Number. The right-most window sets up the port number. Left setup window: defines the module number minus one Right setup window: defines the output port number minus one
Start Byte	byte	0 to length of file	Start byte in file (used when mode is Segment or Sequence)
End Byte	byte	0 to length of file	End byte in file (used when mode is Segment or Sequence)
Cache Pages			Should be set to 4.

Parameter	Unit	Range or Values	Description
Disable MUX Header	Selectable setup	Enable Disable	Don't detect the MUX header. When this option is enabled, it tells the module not to determine if there is a MUX header present on the data buffers. If the operator playing back data that does not have a MUX Header and suspects that the data itself might contain a pattern that matches the MUX Header Sync, then it is good practice to enable this option. When this option is disabled (no check), at module start, the module looks for the MUX Header sync pattern. If it does not detect MUX Header sync patterns, it assumes no MUX Headers present. If no MUX Header sync present at start, then it does not continue to look for MUX Header sync pattern.
Strip MUX Header	Selectable setup	Enable Disable	If enabled, looks to see if MUX header is present and strips it if it is. If disabled, then the MUX header is transmitted with the data.
Play PAUSE	Selectable setup	Enable Disable	Stops playback, leaving file pointer in position. Stops data during the flow. If the desktop is enabled with this configuration parameter set, then data flow does not start until the check is removed.
Play Send One Now	Selectable setup	Enable Disable	"Play Send One Now" allows the operator to output one buffer of data and then stop output. The option can be repeatedly enabled to output each buffer one at a time. This option should be used only when the "Play Pause" button has been enabled. If "Play Pause" is enabled, then the data output has been paused. With data paused, each time the "Play Send One Now" option is enabled, then one buffer of data is output. NOTE that this option does not require the "SEND" to be issued before it is enabled. "Play Send One Now" is enabled as soon as the check is inserted in the setup box.
Ex Log File	Selectable setup	Enable Disable	If enabled, identifies the file as one that was written by Monarch EX board. Monarch EX board provides a data logging capability and the file is written with unique status info appended to each data buffer. Therefore, this option must be enabled if the file being read was generated by the Monarch EX.
StripExStatus	Selectable setup	Enable Disable	Applies to data written by Monarch EX only. If enabled, strips the status information that is automatically appended/written by the Monarch EX.
Buffered Mode	Selectable setup	Enable Disable	When enabled, the module is setup in "Buffered Mode." When disabled, the module is setup in "Non-buffered Mode." Non buffered mode: module caches the data. Faster than buffered mode. Buffered mode: OS caches the data.

Status Window Definition

When you click the Status button on the FILE PLAYBACK VERSION 2 MODULE window, a Status FILE PLAYBACK VERSION 2 window appears, as shown below.



Module 39-4: Status FILE PLAYBACK VERSION 2 MODULE Window

Module 39-5: FILE PLAYBACK VERSION 2 MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
File	text	Reflects configuration	Logical disk, path, and file name that the module is reading (or is trying to read)
Frame size	Bytes	Reflects configuration	Transmitted size. If MUX header present, is data buffer plus MUX header. If no MUX header present, then it is the data buffer size.
Byte Position	Bytes	0 to file size	Displays the current read (pointer) location in file
File Size	Bytes	file size	Displays total byte count of the file to be read back
Loop	Incremental counter	0 to value at time of disable	Displays number of times file has been played when in Loop or Sequence mode
Play	text	Fixed, Segment, Loop, Sequence	Reflects the module setup. It displays the module output mode: Fixed, Segment, Loop, or Sequence modes
State	text	Waiting, Running, Paused, Done	Displays the enabled state of the module, which also reflects setup: "Waiting", "Running", "Paused", or "Done"
Progress bar	Graphic display	Size of the display window	Dots that indicate relative position in the file
Empty Status bar Window	Text display	Open, Play, Pause, Complete	Provides details about module status and configuration

Module 40 FILE RECORDER MODULE

MODULE NAME: FileRecorder.dll

Key Functions

- Logs binary data to hard disk
- Provides configurable modes: fixed file size, dynamic file size that grows with input, and circular file that uses a fixed file size and loops back to file start when file end is reached

Required Hardware

- The FILE RECORDER MODULE does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

Do not exceed the limitation of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module.

CONNECTIONS: Output is to a file.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

The FILE RECORDER MODULE supports logging binary data to hard disk or to a piped device. (*Note: If buffer size value is set to a number that is modulo-4096, data cannot be logged to a piped device.)* The FILE RECORDER MODULE can log the output from any module. For example, the FILE RECORDER MODULE can be used to log frame data from the Serial Input Module, data buffers from a Network Sockets Module, or packets from a Packet Processor Module.

The FILE RECORDER MODULE can log raw frame data or frame data with time stamp and quality annotation.

The FILE RECORDER MODULE supports three record modes. In fixed mode, the recorder logs data until the end of file is reached and then closes the file. In dynamic mode, the recorder logs data until disabled or out of disk space. In circular mode, the recorder logs data until the end of file is reached and then loops back to the start of the file.

Input Data Format

Input data is buffers from any module.

Output Data Format

File Format

If no headers are selected in configuration/setup, then the FILE RECORDER MODULE logs the buffer of data to the file as it is received.

If logging frame data from the Serial Input Module, note that the buffer size for each frame depends on the frame length and the word size.

PCM Word sizes greater than 8 will be right justified in 2-byte words. The binary data is written in Intel little endian format (Low Byte, High Byte). MSB precedes the LSB for each byte. For use with Sun, HP, or SGI workstations, binary data must be byte re-ordered in software prior to processing.

Possible Header Formats

If Record Status, Record Time Stamp, or Record MUX Header are selected, then the File Recorder prepends a header to the input buffer and then writes it to the file as shown in the following diagrams. For the last three diagrams, the MUX header is automatically inserted.

MUX (8 bytes)	***Data Field*** (depends on Frame Length and Word Size)

Module 40-1: File Recorder Format with Record MUX Header enabled

MUX (8 bytes)	***PB-4*** (6 bytes)	***Data Field*** (depends on Frame Length and Word Size)

Module 40-2: File Recorder Format with Record Time Stamp enabled

MUX (8 bytes)	***Appended Status*** (48 bytes)	***Data Field*** (depends on Frame Length and Word Size)

Module 40-3: File Recorder Format with Record Status enabled

MUX (8 bytes)	***PB-4*** (6 bytes)	***Appended Status*** (48 bytes)	***Data Field*** (depends on Frame Length and Word Size)

Module 40-4: File Recorder Format with Record Time Stamp and Record Status enabled

Header Bit Definitions

The MUX Header structure is given in the table below. The PB-4 Time Code format is given in the second table below. The Appended Status Format is defined in Avtec Serial Input Module Section.

Module 40-5: Avtec PTP for Windows MUX Header Format

Item	Field Name	Format & Size	Value
1	Header Synchronization	Unsigned integer (4 bytes)	30030330 hex
2	Source Module	Unsigned integer (1 byte)	Equal to the source module number minus 1. For example, if Module 3 sends a buffer to the file recorder, the Source Module field is equal to 2.
3	Header Types	Unsigned integer (1 byte)	Defines the other header types that follow the MUX Header. 0 = No additional headers 1 = PB-4 Time Code 2 = Serial Input Appended Status 3 = Both PB-4 and Appended Status
4	Next Header Offset	Unsigned integer (2 bytes)	Offset to the next MUX header in the file relative to the end of the current MUX header. Note: that this is a Little Endian representation, the least signigicant byte precedes the most significant byte in the file. For example if the frame length 128 then the Next Header Offset appears in the file as 80 00 hex.
	Total Length	8 bytes	

Module 40-6: NASA PB-4 Time Code Format

Item	Field Name	Format & Size	Value
1	Days of Year	Unsigned integer (11 bits)	Range 1 to 365
2	Milliseconds of Day	Unsigned integer (27 bits)	Range = 0 to 86399999
3	Microseconds of a Millisecond	Unsigned integer (10 bits)	Range = 0 to 999
	Total Length	6 bytes	

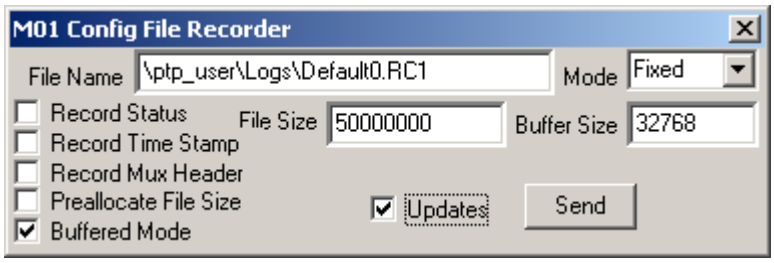
Module Window

When *FileRecorder.dll* is added to a desktop, the File Recorder dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 40-7: FILE RECORDER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure FILE RECORDER MODULE Board Parameters sub-menu.



Module 40-8: Config FILE RECORDER MODULE Window

Module 40-9: FILE RECORDER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
File Name			Defines the log file name
Mode			Defines the recording mode: Fixed, Dynamic, Circular Fixed: To use this mode, the operator defines a FILE SIZE. Based on this file size, the File Recorder module saves data to file until the end of file is reached, and then closes the file. If the data input is more than the file size can hold, the end of the data is lost. Dynamic: In this mode, a file size is not predetermined. Therefore, the recorder logs data to file until either the end of the data flow input, until it is disabled, or until it runs out of disk space. Circular: The Circular mode also uses the defined FILE SIZE. However, in this mode, data is logged to file until either the data flow ends or the recorder is disabled. If the end of the file is reached before the end of the data flow, then the recorder returns to the file start and begins to overwrite data. Therefore, data at the start of the data flow is lost if the file size is too small for the data flow.
File Size			Size of file in bytes
Buffer Size			Defines the buffers size in bytes for file write operations. For best performance, the value should be a multiple of 4096. For best results , the buffer size should also reflect the data flow rate: the slower the rate, the smaller the buffer size should be.
Record Status			When enabled, the Avtec PTP for Windows generated status AND MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Record Time Stamp			If checked, the File Recorder writes the buffer Time Stamp in PB-4 format to the file before the buffer of data. The MUX Header is also written to the file before each buffer of data when Record Time Stamp is checked.

Parameter	Unit	Range or Values	Description
Record MUX Header			If checked, the File Recorder writes the Multiplex (MUX) header to the file before each buffer of data. The MUX header defines which module sent the buffer to the File Recorder. It also defines which other headers (e.g. PB-4 or Appended Status) are present. Useful when logging multiple streams to a single file.
Preallocate File Size			This is an enable/disable field. If enabled, when the desktop is enabled the first thing the enable does is to reserve disk space for the size of the file specified in FILE SIZE field.
Buffered Mode			

Extended Options Window Definition

There is no extended options window for the FILE RECORDER MODULE.

Configuration Notes

When you click the Command button on the FILE RECORDER MODULE window, a Command FILE RECORDER window appears, as shown below.

Module 40-10: Command FILE RECORDER MODULE Window

Module 40-11: FILE RECORDER MODULE Command Window Parameters

Command	Description
Close File	Instructs recording to close. If recording mode is fixed, the file is truncated to the current size
Home File	Instructs the Recorder to seek the start of the file

Status Window Definition

When you click the Status button on the FILE RECORDER MODULE window, a Status FILE RECORDER window appears, as shown below.

Module 40-12: Status FILE RECORDER MODULE Window

Module 40-13: FILE RECORDER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
File Name	text	text	Displays filename in use
Bytes Written	bytes	Incrementing value	Displays bytes written into the file
Text Status Window			Displays a visual cue of the file recording rate and status

Module 41 FILE RECORDER VERSION 2 MODULE

MODULE NAME: FileRecorderVer2.dll

Key Functions

- Records a data stream to a binary file
- Optionally prepends a header, time tag, and data quality annotation to each buffer
- Supports file sizes >4 GB
- Optimized for dual processor systems (PTP-EX)
- Accepts input from multiple source modules
- Optionally provides a spooler capability

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of FILE RECORDER VER2 MODULES may be loaded in a desktop. The maximum number of modules allowed in a desktop is 32 and CPU usage should be monitored since this is a software module. If multiple high-speed data streams are being recorded, each file recorder should be configured to write data to its own physical disk.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide data input to this module.

CONNECTIONS: Output from this module is to the selected disk via the Windows file system. If operator chooses to use this module as a spooler (event input required), then data that is written to disk is then spooled to port one for output at the input event rate.

EVENTS: Does not generate an event output. Can accept event input if operator chooses to use this module as a spooler.

Module Functional Description

The FILE RECORDER VER2 MODULE supports logging binary data to hard disk or to a piped device. The FILE RECORDER VER2 MODULE can log the output from any module. For example, the File Recorder can be used to log frame data from the Serial Input Module, data buffers from a Network Sockets Module, or packets from a Packet Processor Module. It can log raw frame data or frame data with time stamp and quality annotation. The File Recorder configuration parameters, status display, and file format are defined below.

The FILE RECORDER VER2 MODULE is extension of the original file playback module. The Ver2 enhancements include the ability to support file sizes greater than 4 GB and improved performance on dual processor machines.

The Recorder supports three record modes. In fixed mode, the recorder logs data until the end of file is reached and then closes the file. In dynamic mode, the recorder logs data until disabled or out of disk space. In circular mode, the recorder logs data until the end of file is reached and then loops back to the start of the file. The recorder can log raw frame data or frame data with time stamp and quality annotation.

If the operator chooses, this module can be used as a spooler. To use the spooler function, the operator simply needs to provide an event input. The module automatically strips any MUX headers if used as a spooler.

NOTE: If input is from multiple sources, the MUX header must be written with each buffer or there is no way to track which module has provided each data buffer.

Input Data Format

Any data format can be written to disk.

File Recorder File Format

If no headers are selected, then the File Recorder logs the buffer of data to the file as it is received. If logging frame data from the Serial Input, note that the buffer size for each frame depends on the frame length and the word size. PCM Word sizes greater than 8 will be right justified in 2-byte words. The binary data is written in Intel little endian format (Low Byte, High Byte). MSB precedes the LSB for each byte. For use with Sun, HP, or SGI workstations, binary data must be byte re-ordered in software prior to processing.

If Record Status, Record Time Stamp, or Record MUX Header is selected, then the File Recorder prepends a header to the input buffer and then writes it to the file as shown below. For the last three File Recorder formats, the MUX header is automatically inserted.

For accuracy, the AV_PTP.h header file should always be checked for MUX Header and MUX Header Ver 2 format. The following tables should be considered a general guideline.

Based on the MUX Header format, a new MUX Header has been defined: MUX Header Version 2. MUX Header Ver2 adds a 4-byte field that defines module output port number, and it is located between Source Module and Header Types. Version 2 also adds pad for future expansion. After Header Types, there is a pad of 32 bytes. To differentiate from the original header, MUX Header Ver 2 has a new sync pattern: 30030331 hex.

MUX	***Data Field***
(8 bytes)	(depends on Frame Length and Word Size)

Module 41-1: File Recorder Format with Record MUX Header enabled

MUX	***PB-4***	***Data Field***
(8 bytes)	(6 bytes)	(depends on Frame Length and Word Size)

Module 41-2: File Recorder Format with Record Time Stamp enabled

MUX	***Appended Status***	***Data Field***
(8 bytes)	(48 bytes)	(depends on Frame Length and Word Size)

Module 41-3: File Recorder Format with Record Status enabled

MUX	***PB-4***	***Appended Status***	***Data Field***
(8 bytes)	(6 bytes)	(48 bytes)	(depends on Frame Length and Word Size)

Module 41-4: File Recorder Format with Record Time Stamp and Record Status enabled

The MUX Header structure is given in the table below.

Module 41-5: MUX Header Format

Item	Field Name	Format & Size	Value
1	Header Synchronization	Unsigned integer (4 bytes)	30030330 hex
2	Source Module	Unsigned integer (1 byte)	Equal to the source module number minus 1. For example, if Module 3 sends a buffer to the file recorder, the Source Module field is equal to 2.
3	Header Types	Unsigned integer (1 byte)	Defines the other header types that follow the MUX Header. 0 = No additional headers 1 = PB-4 Time Code 2 = Serial Input Appended Status 3 = Both PB-4 and Appended Status
4	Next Header Offset	Unsigned integer (2 bytes)	Offset to the next MUX header in the file relative to the end of the current MUX header. Note that this is a Little Endian representation, the least significant byte precedes the most significant byte in the file. For example if the frame length 128 then the Next Header Offset appears in the file as 80 00 hex.
	Total Length	8 bytes	

The PB-4 Time Code format is given in the table below.

Module 41-6: NASA PB-4 Time Code Format

Item	Field Name	Format & Size	Value
1	Days of Year	Unsigned integer (11 bits)	Range 1 to 365
2	Milliseconds of Day	Unsigned integer (27 bits)	Range = 0 to 86399999

Item	Field Name	Format & Size	Value
3	Microseconds of a Millisecond	Unsigned integer (10 bits)	Range = 0 to 999
	Total Length	6 bytes	

The Appended Status Format is defined in Avtec Serial Input Module Section of the Avtec PTP for Windows User's Guide.

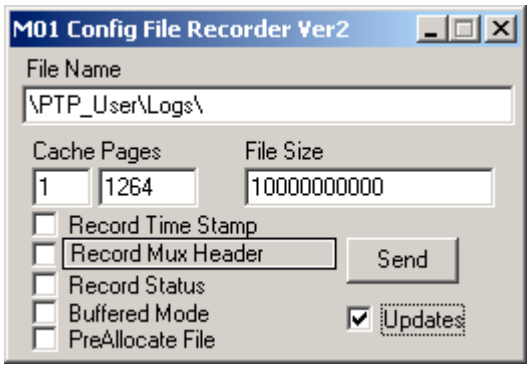
Module Window

When *FileRecorderVer2.dll* is added to a desktop, the File Recorder Version 2 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 41-7: FILE RECORDER VER2 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below**.** The table that follows defines the fields in the Configure FILE RECORDER VER2 MODULE Parameters sub-menu.



Module 41-8: Config FILE RECORDER VER2 MODULE Window

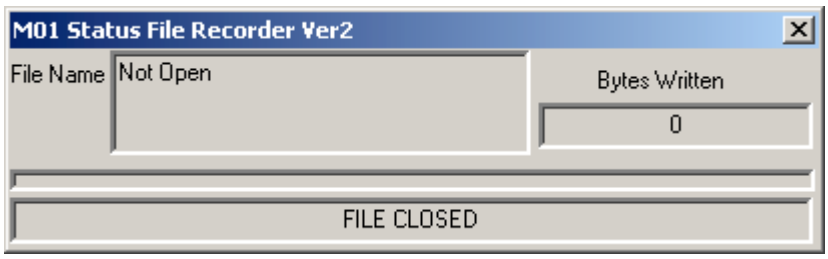
Module 41-9: FILE RECORDER VER2 MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
File Name	text	Directory path that exists in system	The user enters the file name that will be used for data logging/reading. Name must include path. If the file already exists, it will automatically be overwritten at run time without any warnings. If the file does not exist, it will be created at enable. If the file path entered here does not exist, then the module creates the directories and file name as long as the logical disk exists.

Parameter	Unit	Range or Values	Description
File Size	bytes	Limited by disk space	Defines the expected file size in bytes. Should always be set up to be at least the size of the expected amount of data input. If this size is exceeded data is lost. The module pre-allocates a file of this size when enabled so there must be enough disk space available. The file size will be set to the number of bytes actually recorded when the module is disabled.
Cache	integer	1 to 64	This field defines the number of cache pages. If a number greater than 64 is entered, then this field automatically sets itself back to 64. Cache Page size is fixed at 4096 bytes. Total Cache Page size (number of pages * 4096) is the amount of data that is buffered before the module writes to the disk. The module WILL write any partial buffers of data to disk on disable.
Pages	integer	1264	Display value only and the number in the window has no relevance.
Record Time Stamp	Selectable Setup State	Enable Disable	When enabled, the Avtec PTP for Windows generated time stamp AND MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Record MUX Header **Record MUX Header Ver2**	Selectable Setup State	Enable Disable	NOTE that this is both an enable/disable (check box selected is enabled) and a two-state button. If enabled, then the configuration checks the state of the button. The button setup is changed by selecting it. The button options are "Record MUX Header" and "Record MUX Header Ver 2." When enabled, the Avtec PTP for Windows generated MUX header is recorded with each data unit written to file. Depending on the button setup, a different MUX Header is written to file.
Record Status	Selectable Setup State	Enable Disable	When enabled, the Avtec PTP for Windows generated status AND MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Buffered Mode	Selectable Setup State	Enable Disable	If enabled, lets file system handle caching. Data is first copied to memory, then file system thread later sends copied data to disk. Buffered mode must be used to record over the network. If disabled, disables file system cache and writes data directly to disk. Must write in multiples of cluster size. Non-buffered mode is faster.
PreAllocate File	Selectable Setup State	Enable Disable	Gives a file size and claims that space upfront. Must preallocate on enabling the module, so the module may take a long time to start. The advantage is minimizing head movements.

Status Window Definition

When you click the Status button on the FILE RECORDER VER2 MODULE window, a Status FILE RECORDER VER2 window appears, as shown below.



Module 41-10: Status FILE RECORDER VER2 Window

Module 41-11: FILE RECORDER VER2 MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
File Name	Text	Reflects the filename setup in config window	Displays filename in use. Unable to Open File Creates the path if the physical/logical drive exists
Bytes Written	Bytes	Reflects the data flow input bytes/max is available disk space	Displays bytes written to file.
Progress Indicator	Graphic	Length of the window	Displays a visual cue of the spooling rate and status.
Status summary window	Text	See description	Gives the status of the file identified by name in the configuration window: File Open: file is open and ready to write data. File Closed: This status occurs if operator issued closed command, disabled, or in fixed mode it has reach the max file size File Active: Status appears when data is first written to file and does not go away until file is closed File Full/File Closed: This indicates for fixed mode only that the setup file size has been exceeded and data is being dropped from that point forward Overflow Error: Data is being input faster than the disk can handle. Changing the buffer size to a bigger size may mitigate the problem. If this is true, then the issue was CPU usage. If the hard drive is the bottleneck, then changing the buffer size does not fix the problem.

Module 42 FILE SPOOLER MODULE

MODULE NAME: Spooler.dll

Key Functions

- Provides rate buffering of a data stream
- Simultaneously writes data to and reads back from a file
- Optionally writes Avtec PTP for Windows generated status and time with each frame to file
- Maintains data in file until file is overwritten
- Buffers data for output to a socket

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of FILE SPOOLER MODULES may be loaded in a desktop. Limitations are system imposed. The maximum number of modules allowed in a desktop is 32 and CPU usage should be monitored since this is a software module.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide data input to this module.

CONNECTIONS: Output is to port one.

EVENTS: Like the File Playback Module, this module requires an event input to drive data output. Module outputs one data buffer whenever it receives an event input. If using a Network Sockets Module (Sockets or Sockets Version 2) as the output, only the Sockets Module (not the Sockets Version 2) can be used to send events to this module. When using the Sockets Module, if the network goes down, the File Spooler will continue spooling to the local disk, then dumps the spooled data to the network when it returns

Module Functional Description

The FILE SPOOLER MODULE combines capabilities from both the File Recorder and File Playback Modules. The Spooler takes data from one module (i.e. Avtec Serial Input), spools the data into a file on the local hard drive or a piped device, and plays back the data to another module (i.e. Network Sockets) at a different (faster or slower) rate. Similar to the Playback Module, the Spooler Module requires an event on output to drive the playback.

Currently, the output frame length of the FILE SPOOLER MODULE must be an integer multiple of the incoming frame size (i.e. 1x, 2x, etc.).

If data buffers are written to file with the MUX header (this is defined by the setup), then data output (when an event input is received) is based on information contained in the MUX header (refer to File Recorder Module description for details of MUX and status header).

Input Data Format

Any data format can be written to disk.

Output Data Format

If data was written to file with a MUX header, data is always output according to the length in the MUX header. This means that each event input to the module causes output of one data unit the length defined by the MUX header. Data is output until the end of file is reached. When spooling data that was written with the MUX header, if the buffer size defined by configuration is greater than the size of data in MUX header, then data may be stuck at the end of a data flow and not output from the module. In this case, the last buffer of data is not written to file (and therefore is not available for output) until the file is closed. Setting the configuration to automatic buffer size (1) eliminates this problem.

If data was written to file without a MUX header, then each event input to the module causes the output of one buffer of data of the size requested by the receiving module.

See the configuration notes for this module for more information on options for using this module that include setting up prioritized playback.

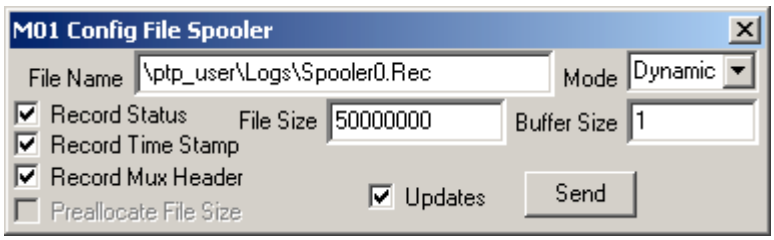
Module Window

When *Spooler.dll* is added to a desktop, the File Spooler dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 42-1: FILE SPOOLER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure FILE SPOOLER MODULE Board Parameters sub-menu.



Module 42-2: Config FILE SPOOLER MODULE Window

Module 42-3: FILE SPOOLER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
File Name	text	Directory path that exists in system	The user enters the file name that will be used for data logging/reading. Name must include path. If the file already exists, it will automatically be overwritten at run time without any warnings. If the file does not exist, it will be created at enable. If the file path entered here does not exist, then the module creates the directories and file name as long as the logical disk exists.
Mode	Selectable Setup State	Fixed Dynamic	Allows the operator to either limit the file size that can be written (fixed) or set the system up to expand the file until all data is written (dynamic). The file size used is the size entered in the File Size config field (listed below). Fixed file size should be run with caution, if file size is exceeded, then file is closed as soon as the limit is reached and the rest of data is dropped.
Record Status	Selectable Setup State	Enable Disable	When enabled, the Avtec PTP for Windows generated status AND MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Record Time Stamp	Selectable Setup State	Enable Disable	When enabled, the Avtec PTP for Windows generated time stamp AND MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Record MUX Header	Selectable Setup State	Enable Disable	When enabled, the Avtec PTP for Windows generated MUX header is recorded with each data unit written to file. Check box setup; insert a check to enable function.
Preallocate File Size	N/A	N/A	Not an option. It is always off. Historically, this field allowed the option to be on, which pre-allocates space for the file size set up at enable. If additional disk space was required, then the module requested the space for each buffer input after the file size was exceeded.
File Size	bytes	Limited by disk space	Defines the expected file size in bytes. Should always be set up to be at least the size of the expected amount of data input. In fixed mode, if this size is exceeded data is lost. In dynamic mode, file size has no relevance since it writes only the amount of data input. This means that in Dynamic Mode, a large file size number is not a problem because it only records the actual size of data written.

Parameter	Unit	Range or Values	Description
Buffer Size	bytes		Defines the buffer size for data input to the module. This is the amount of data that is buffered before the module writes to the disk. Should be multiple of 4096 for the best performance. Faster rates do require bigger buffer sizes. Set to 1 for automatic buffer sizing. Module outputs at the size requested by the module that generates the event, or if data written with MUX header, then output size is always the size recorded in the MUX header.

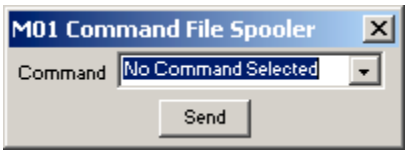
Extended Options Window Definition

There is no extended options window for the FILE SPOOLER MODULE.

Configuration Notes

Multiple FILE SPOOLER MODULES can be chained to create a prioritized playback. To do so, a chain of events must be setup. First, multiple FILE SPOOLER MODULES are loaded, the number of which is determined by the number of channels required for playback. The highest priority File Spooler channel to playback (priority 1) requires an event connection from an output module (i.e. Avtec Serial Output or Network Sockets). The priority 2 Spooler then requires an event connection from the priority 1 Spooler, and so on from Spooler to Spooler. This chaining of events determines the priority levels of playback.

The FILE SPOOLER MODULE includes a Command window that is defined in the following figure and table definitions.



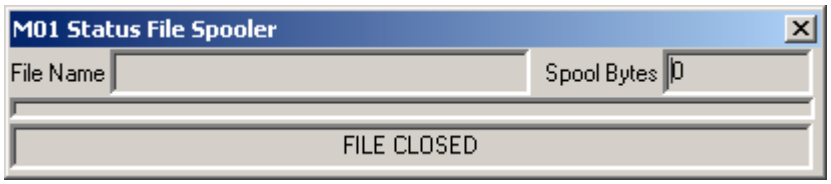
Module 42-4: Command FILE SPOOLER MODULE Window

Module 42-5: FILE SPOOLER MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
Close File	command	N/A	Activates at SEND. Stops data from being written to disk. Close file allows the user to change file names in the middle of a data flow. A file name change during a data flow does not become valid until a close file has been issued.
Home File	command	N/A	Activates at SEND. Instructs spooler to seek start of file. Sets input and output pointers back to the file beginning. Existing file data get overwritten.

Status Window Definition

When you click the Status button on the FILE SPOOLER MODULE window, a Status FILE SPOOLER window appears, as shown below.



Module 42-6: Status FILE SPOOLER MODULE Window

Module 42-7: FILE SPOOLER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
File Name	Text	Reflects the filename setup in config window	Displays filename in use. Unable to Open File Creates the path if the physical/logical drive exists
Spool Bytes	Bytes		Displays bytes written to file.
Progress Indicator	Graphic		Displays a visual cue of the spooling rate and status.
Status summary window	Text	See description	Gives the status of the file identified by name in the configuration window: File Open: file is open and ready to write data. File Closed: This status occurs if operator issued closed command, disabled, or in fixed mode it has reach the max file size File in Loop: File has reached the end of the file at least once and has restarted at the beginning-- should only ever occur if system is being set up remotely. It applies only to Loop Mode, which can only be set up remotely. It is not an option for local configuration. File Active: Status appears when data is first written to file and does not go away until file is closed File Expanding: This status shows any time it goes past file size that was set up. File Full/File Closed: This indicates for fixed mode only that the setup file size has been exceeded and data is being dropped from that point forward Overflow Error: Data is being input faster than the disk can handle. Changing the buffer size to a bigger size may mitigate the problem. If this is true, then the issue was CPU usage. If the hard drive is the bottleneck, then changing the buffer size does not fix the problem.

Module 43 GATHER SCATTER MODULE

MODULE NAME: GatherScatter.dll

Key Functions

- Allows the user to send a contiguous portion of a buffer on to another module.
- Accepts any input data format
- Extracts data from buffer based on configurable offset and data length

Required Hardware

- The GATHER SCATTER MODULE requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of GATHER SCATTER MODULES can be added to a desktop as long as the 32-module limit per desktop is not exceeded.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives buffers from other modules loaded in an Avtec PTP for Windows desktop.

CONNECTIONS: Output port 1 must be connected to another module that is designed to receive the stripped portion of data.

EVENTS: No events are generated or expected.

Module Functional Description

The Gather Scatter Module is used to extract user-specified bytes from a telemetry frame/packet, discarding the remaining data, and passing only the extracted bytes. The user-specified bytes are contiguous within the buffer. The chosen bytes are placed at the beginning of the output buffer.

Input Data Format

Any input buffer of data is permitted. No assumptions are made as to data content.

Output Data Format

No assumptions or validation of the output data are made. The data buffer is simply treated as bytes.

Module Window

When GatherScatter.dll is added to a desktop, the Gather Scatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 43-1: GATHER SCATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure GATHER SCATTER MODULE Board Parameters sub-menu.

Module 43-2: Config GATHER SCATTER MODULE Window

Module 43-3: GATHER SCATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Offset	bytes	0 to the end of the buffer	Location of first byte to be extracted within frame/packet. The most significant bytes is offset zero (0).
Length	bytes	1 to the number of bytes in the buffer	Number of bytes from Offset to extract from frame/packet.

Extended Options Window Definition

There is no extended options window for the GATHER SCATTER MODULE.

Configuration Notes

No validation of the offset or length parameters is made. It is possible that the user could specify an offset greater than the actual length of the data in the buffer. It is also possible that the user could specify an offset within his data and a length that causes garbage (in memory after his data) to be included in the output buffer.

Status Window Definition

There is no status window for the GATHER SCATTER MODULE.

Module 44 GDP PSK MODULATOR MODULE

MODULE NAME: PSKMod.dll

Key Functions

- Controls GDP PSK Modulator

Required Hardware

- General Data Products PSK Modulator

Add Module Guidelines

Use one module per card to be used.

Required Input Source, Connections, and Events

INPUT SOURCE: All data is received from cables plugged into the back of the card.

CONNECTIONS: No module connections are made with this module.

EVENTS: No events are generated or required.

Module Functional Description

The PSK Modulator supports the GDP PSK004 PSK Modulator card. The PSK Modulator module does not have any data inputs or outputs or event outputs and cannot be "connected" to another module. The PSK Modulator board generates a modulated subcarrier according to user inputs. This card does not report any status.

Input Data Format

Baseband data.

Output Data Format

PSK Modulated data.

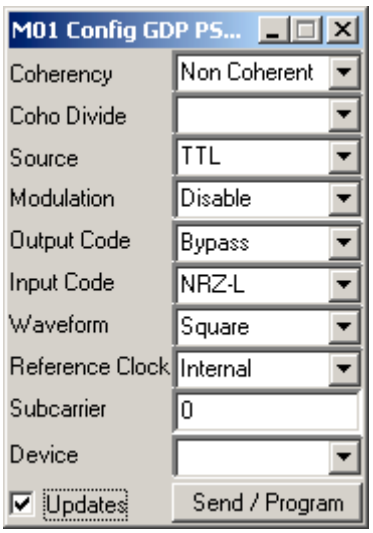
Module Window

When PSKMod.dll is added to a desktop, the GDP PSK Modulator dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 44-1: GDP PSK MODULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure GDP PSK MODULATOR MODULE Board Parameters sub-menu.



Module 44-2: Config GDP PSK MODULATOR MODULE Window

Module 44-3: GDP PSK MODULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Coherency		Coherent Non Coherent	Choice of Coherent or Non-coherent operation
Coho Divide		2, 4, 8, 16, 32, 64, 128	Coherent Divisor; choice of 2, 4, 8, 16, 32, 64, 128
Source		TTL RS-422	Allows either TTL or 422 input
Modulation		Disable Enable	Choice of Enable or Disable
Output Code		NRZ-L, NRZ-M, NRZ-S, Bypass	Choice of NRZ-L, M, S, or Bypass

Parameter	Unit	Range or Values	Description
Input Code		NRZ-L, NRZ-M, NRZ-S, BIO-L, BIO-M, BIO-S, Bypass	Choice of NRZ-L, M, S, or BI0-L, M, S, or Bypass
Waveform		Square Sine	Choice of Sinewave or Squarewave
Reference Clock		Internal, Ext 5 MHz, Ext 10 MHz	Choice of Internal, or External 5 or 10 MHz
Subcarrier	Hz		Subcarrier Frequency in Hz
Device		Select Device	Selects which device is to be controlled. Devices are defined in AUXIO.

Extended Options Window Definition

There is no extended options window for the GDP PSK MODULATOR MODULE.

Configuration Notes

When you click the Command button on the GDP PSK MODULATOR MODULE window, a Command GDP PSK MODULATOR window appears, as shown below.

Module 44-4: Command GDP PSK MODULATOR MODULE Window

Module 44-5: GDP PSK MODULATOR MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected			No command selected. No command is sent to the board.
Disable Subcarrier			Disables the subcarrier by setting frequency to 0
Enable Subcarrier			Enable the subcarrier by setting frequency to programmed value

Status Window Definition

There is no status window for the GDP PSK MODULATOR MODULE.

Module 45 IN LINE UNZIP MODULE

MODULE NAME: InLineUNZip.dll

Key Functions

- Unzips (decompresses) data compressed by the In-Line Zip Module
- Outputs decompressed data
- Decompresses on a per input buffer boundary
- Decompresses using PK Unzip libraries

Required Hardware

- This module does not require any special hardware to be installed in the Avtec PTP for Windows system (does requires third party software).

Add Module Guidelines

Any number of IN-LINE UNZIP MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data

Required Input Source, Connections, and Events

INPUT SOURCE: Receives input from any other module.

CONNECTIONS: Outputs to port one, which must be connected to another modules input or the data is lost.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

The IN-LINE UNZIP MODULE provides on-the-fly data decompression of data compressed using the In-Line Zip Module. Compressed data is input to the module, and decompressed data is output to another module. There is no configuration or command window associated with this module.

The IN-LINE UNZIP MODULE decompresses on a per buffer boundary. The buffer boundary must match the boundary used by the In-Line Zip Module.

Uses third party software located in ptpnt/aim32.dll.

Input Data Format

This module expects input buffers that have each been compressed by the In-Line Zip Module.

Output Data Format

Outputs decompressed data.

Module Window

When InLineUNZip.dll is added to a desktop, the In Line UnZip dialog appears. Left mouse click Status to view current module state and data processing statistics.

Module 45-1: IN LINE UNZIP MODULE Window

Configure Window Definition

There is no configuration window for the INLINE UNZIP MODULE.

Extended Options Window Definition

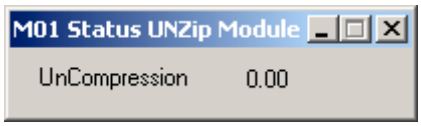
There is no extended options window for the INLINE UNZIP MODULE.

Configuration Notes

There are no configuration notes.

Status Window Definition

When you click the Status button on the INLINE UNZIP MODULE window, a Status UNZIP window appears, as shown below.



Module 45-2: Status IN LINE UNZIP MODULE Window

Module 45-3: IN LINE UNZIP MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
UnCompression	percentage	0 to 100	Displays the average decompression ratio. Not calculated on a per buffer boundary, but reflects an average. Calculated using bytes output divided by bytes input

Module 46 INLINE ZIP MODULE

MODULE NAME: InLineZip.dll

Key Functions

- Compresses input on a per buffer boundary
- Outputs compressed buffers to port one
- Uses PK zip library

Required Hardware

- This module does not require any special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of INLINE ZIP MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Receives input from any other module.

CONNECTIONS: Outputs to port one, which must be connected to the input of another module or the data is dropped.

EVENTS: Neither generates an event or receives an event input.

Module Functional Description

The INLINE ZIP MODULE provides on-the-fly data compression. A serial stream is input to this module and a compressed stream is output to another module. There is no configuration or command window associated with this module.

The INLINE ZIP MODULE compresses on a per buffer boundary. Each input buffer is compressed and then output. As a result all processing after this module must be buffer aligned. The buffer can not be broken up because the unzip works on the same boundaries. PKZip is used to compress.

This module relies on third party software located in ptpnt/aim32.dll, which is a pk zip and unzip library.

Data compressed by this module can be decompressed with the InLine UnZip Module or a PK unzip library can be used.

Input Data Format

Any data format can be input. The INLINE ZIP MODULE expects buffers of data input. Depending on the source module, the buffers may be on a frame or packet boundary, but the module processes on a buffer boundary and the data content is irrelevant.

Output Data Format

Outputs compressed data.

Module Window

When InLineZip.dll is added to a desktop, the In Line Zip dialog appears. Left mouse click Status to view current module state and data processing statistics.

Module 46-1: INLINEZIP MODULE Window

Configure Window Definition

There is no configuration window for the INLINE ZIP MODULE.

Extended Options Window Definition

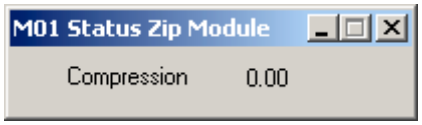
There is no extended options window for the INLINE ZIP MODULE.

Configuration Notes

There are no configuration notes.

Status Window Definition

When you click the Status button on the INLINE ZIP MODULE window, a Status ZIP window appears, as shown below.



Module 46-2: Status INLINEZIP MODULE Window

Module 46-3: INLINEZIP MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Compression	percentage	0 to 100	Displays the average compression. Does not report the per buffer compression, but reports the average compression for all buffer input. The statistic is the number of bytes received divided by the number of bytes transmitted.

Module 47 IPDU FORMATTER MODULE

MODULE NAME: IPDUFormatter.dll

Key Functions

- Appends 32-byte IPDU header to each received frame (buffer) and outputs IPDU packet
- Provides synchronization pattern and packet size (header and frame) that can be used for network transmissions
- Automatically reads status information and inserts into header for CCSDS frames that were processed by Serial Input or VCP Modules
- Encapsulates any frame format for output (provides status in header for CCSDS format input only)

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the Avtec PTP for Windows-defined 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Can receive data from any module. If all status fields in the header are required, then the data source needs to be a VC Processor (status fields are not necessary to use the header format).

CONNECTIONS: Outputs to port one, and the output must be connected to another module's input or the data is lost.

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The IPDU FORMATTER MODULE is an encapsulation module: it received input and transmits the unchanged-data with an IPDU header prepended. The format was specifically designed for transmission across the network. The IPDU header format is defined in **Appendix B**. The IPDU FORMATTER MODULE output is defined as an IPDU packet (header and data).

The IPDU Header is a 32-byte header that includes a synchronization mark and a message length (including header). It is highly recommended for network traffic because the synchronization mark and length make it easy to correct for network drops or problems at the receive side.

The IPDU FORMATTER MODULE typically receives telemetry frames with quality annotation from the Serial Input Module or the VCP Module. The IPDU FORMATTER MODULE is typically connected to a Socket Module for network output. The IPDU FORMATTER MODULE

could also be used in an operations center to encapsulate commands for network transmissions.

Input Data Format

Any data input is acceptable. Some fields of the IPDU Header require CCSDS frames input to complete the status fields, but the header is still very useful for TDM data or other non-CCSDS formats.

Output Data Format

Output is the input data with the IPDU header appended.

Module Window

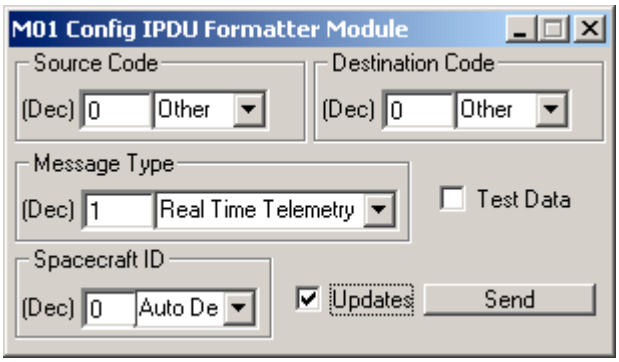
When *IPDUFormatter.dll* is added to a desktop, the IPDU Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. The module does not include a Status or Command Window.

Module 47-1: IPDU FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure IPDU FORMATTER MODULE Board Parameters sub-menu.

The Config IPDU Formatter is used to set user-programmable fields within the IPDU header. The other fields in the IPDU header such as RS data quality and time stamp are filled with information taken from each buffer on a frame-by-frame basis.



Module 47-2: Config IPDU FORMATTER MODULE Window

Module 47-3: IPDU FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
All information Provided in the Config window defines the IPDU header and the information defined here is inserted into the IPDU header that is prepended to each output frame			
SOURCE CODE			
(Dec)	decimal	8-bits	Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit source code in this window that is stamped into each IPDU Header. For any other choice than "other," this window provides the decimal number associated with the selected source. The pull down menu provides the text representation of the most common source codes. When the source is selected and the Send is issued, this window automatically shows the numeric representation for that source-- this is the number that is stamped into each IPDU Header
Pull down menu	Selectable Setup	L7M SFS ASK SPZ WSC WIL VLF VAF WFF other	Provides options for the source code that is stamped into each IPDU header. The operator may select the text representation of a source code, or choose to define a source code by selecting other. Use the text window (Dec) for other definitions. If a text source code is selected, then the numeric representation of the selected source shows in the text window when the SEND is issued.
DESTINATION CODE			
(Dec)	decimal	8-bits	Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit destination code in this window that is stamped into each IPDU Header. For any other choice than "other," this window provides the decimal number associated with the selected destination. The pull down menu provides the text representation of the most common destination codes. When the destination is selected and the Send is issued, this window automatically shows the numeric representation for that destination -- this is the number that is stamped into each IPDU Header
Pull down menu	Selectable Setup	L7M SFS ASK SPZ WSC WIL VLF VAF WFF other	Provides options for the destination code that is stamped into each IPDU header. The operator may select the text representation of a destination code, or choose to define a destination code by selecting other. Use the text window (Dec) for other definitions. If a text destination code is selected, then the numeric representation of the selected destination shows in the text window when the SEND is issued.
MESSAGE TYPE			

Parameter	Unit	Range or Values	Description
(Dec)	decimal	8-bits	Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit message type in this window that is stamped into each IPDU Header. For any other choice than "other," this window provides the decimal number associated with the selected message type. The pull down menu provides the text representation of the most common message types. When the message type is selected and the Send is issued, this window automatically shows the numeric representation for that message type -- this is the number that is stamped into each IPDU Header
Pull down menu	Selectable Setup	Real Time Telem, Playback Telem, Real Time Command, Command Echo, other	Provides options for the message type that is stamped into each IPDU header. The operator may select the text representation of a message type, or choose to define a message type by selecting "other." Use the text window (Dec) for other definitions. If a text message type is selected, then the numeric representation of the selected message type shows in the text window when the SEND is issued.
SPACECRAFT ID			
(Dec)	Decimal	8 bits	Reflects the choice from pull down menu, OR allows the user to define a Spacecraft ID when "other" is selected from the pull down menu.
Pull down menu	Selectable Setup	Auto Detect, LSAT7, XTE, other	Provides Spacecraft ID choices: Auto Detect, LSAT7, XTE, or Other (with value specified in decimal) If set to Auto Detect and data is CCSDS version 1 or version 2 frames, then it reads the SCID from the header and inserts this value into the IPDU field.
OTHER SETUP			
Test Data	Selectable Setup	Enable Disable	Defines the IPDU header test data indicator flag. A single bit in the IPDU header that can be turned on or off. If enabled, then the bit is set in every IPDU header output. If disabled, the bit is zero in every IPDU header output. By definition, 0=operational data and 1=test data. Enter a check to enable (set bit to one).

Extended Options Window Definition

There is no extended options window for the IPDU FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the IPDU FORMATTER MODULE.

Status Window Definition

There is no status window for the IPDU FORMATTER MODULE.

Module 48 IPDU RECEIVER MODULE

MODULE NAME: IPDUReceiver.dll

Key Functions

- Accepts IPDU packet input (frame with 32-byte IPDU header)
- Outputs to data path (uplink) on port 1 (IPDU header optionally stripped)
- Provides echo path on port 2 (optionally swaps source and destination codes)
- Filters all input based on destination code
- Optionally routes data to echo path based on test bit
- Optionally filters port one output based on data type (provides forward command throughput only)
- Synchronizes to sync pattern and packet length, which are both contained in IPDU header

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the Avtec PTP for Windows 32 modules per desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Receives data from socket.

CONNECTIONS: Outputs accepted IPDU packets to port one. Outputs echoed data to port two. If another module is not connected to the output port, than the data on that port is dropped.

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The IPDU RECEIVER MODULE is a socket module that expects IPDU packet input, provides filtering criteria, and includes an echo path. It receives IPDU packets from a network socket connection, which are frames with the 32-byte IPDU header appended, and outputs data to port one and echoes to port two.

The IPDU RECEIVER MODULE detects the start of an IPDU packet using the IPDU header sync pattern. The IPDU RECEIVER MODULE filters and processes the packet based on the IPDU header information and the module configuration. The IPDU module has two data output ports. Port 1 outputs accepted packets with or without the IPDU header. Port 2 is used for command echoes. Echoes are accepted packets with the source and destination code swapped for transmission back to the control center.

The module configuration provides several filtering criteria that can optionally be used. If the filtering is enabled, the order of filtering is as follows: destination code, test bit steering, and message type (data filter that routes to port 1 and 2).

See **Appendix B** for a detailed overview of the IPDU format.

Input Data Format

Socket input must be in IPDU format (data with IPDU header prepended).

Output Data Format

Output on port one may or may not include the IPDU header depending on setup. Echo on port two always includes the IPDU header.

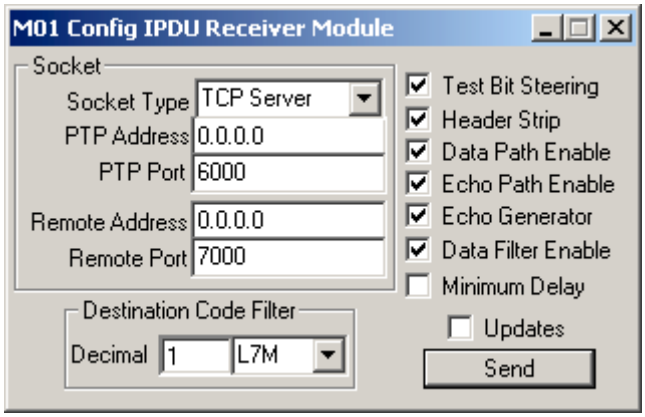
Module Window

When the *IPDUReceiver.dll* is added to a desktop, the IPDU Receiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 48-1: IPDU RECEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure IPDU RECEIVER MODULE Board Parameters sub-menu.



Module 48-2: Config IPDU RECEIVER MODULE Window

Module 48-3: IPDU RECEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
For every setup configuration, an incoming IPDU Packet must pass destination code filtering before it is either output or it goes through any other filtering criteria NOTE: Config window comes up with a default configuration that is the default ground station uplink configuration			
SOCKET			
Socket Type	Selectable Setup	UDP TCP Server TCP Client	Sets the socket type: Disabled, UDP, TCP Client, or TCP Server
PTP Address	Address	IP address	Local Avtec PTP for Windows address. Address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	Port	1 to 65,535	The local socket that is to be opened: socket number.
Remote Address	Address	IP address	Remote IP address. Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal.
Remote Port	Port	1 to 65,535	Remote Socket number -- Defines the socket on the remote machine on which it expects data transfer.
DESTINATION CODE FILTER			
Decimal	decimal	8-bit field	Defines destination code filter: the value that must be contained in an IPDU header destination code field or it is not accepted. Can be used in one of two ways: 1) Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit source code in this window that is used for the filter criteria. 2) For any other choice than "other," this window provides the decimal number associated with the selected source. The pull down menu provides the text representation of the most common source codes. When the source is selected and the Send is issued, this window automatically shows the numeric representation for that source-- this is the number that is used for the filter criteria.

Parameter	Unit	Range or Values	Description
Pull down menu	Selectable Setup	Disabled L7M, SFS, ASK, SPZ, WSC, WIL, VLF, VAF, WFF, or Other	Defines destination code filter: the value that must be contained in an IPDU header destination code field or it is not accepted. Provides options for the source code filter that must match each IPDU header. The operator may select the text representation of a source code, or choose to define a source code by selecting other. Use the text window (Dec) for other definitions. If a text source code is selected, then the numeric representation of the selected source shows in the text window when the SEND is issued.
OTHER SETUP			
Test Bit Steering	Selectable Setup	Enable Disable	Determines if the module checks for test bit set in the IPDU header. If enabled, and the incoming header has the test bit set, the data will not be uplinked, it is sent out port 2 and if the Echo Generator is set, it is treated like an echo If disabled, module does not check for test bit.
Header Strip	Selectable Setup	Enable Disable	Determines if data sent out data port (1) includes the IPDU header or if the header is stripped prior to output. If Enabled, module strips header prior to Data output. If Disabled, data is throughput with the header appended.
Data Path Enable	Selectable Setup	Enable Disable	Determines if data is output on the uplink path (port 1). If enabled, data is output. If disabled, no data is uplinked (no output).
Echo Path Enable	Selectable Setup	Enable Disable	Determines if data is output on the echo path (port 2). If enabled, data is always output on port 2. If the echo generator is also enabled, then each received IPDU packet is output in echo format (source and destination swapped). If the echo generator is not enabled, data is throughput with no changes. If disabled, no data is output on port 2. Insert check to enable.
Echo Generator	Selectable Setup	Enable Disable	When echo path is enabled, this switch determines if data is output as an echo. If enabled (and echo path enabled), outputs each forward command as an echo– swaps destination and source code in header and sends command echo. If disabled (and echo path enabled), then data output port 2 as it is received (no header changes). Insert check to enable.

Parameter	Unit	Range or Values	Description
Data Filter Enable	Selectable Setup	Enable Disable	Determines if the data type field is checked and data filtered according to the data field value. Allows the operator to output only commands on data port 1, which is the uplink path. (If test bit steering is enabled, that logic is applied before data filter; therefore, the data filter logic is never applied). When enabled, works as a Command filter-- determines if it is an uplink/forward command and passes only commands out Data Port 1. By default, the filter is message type 3. So if enabled, the only thing accepted into uplink is type 3, which is a forward command If disabled, does not filter on data type field. Should disable if module is being used for telemetry.
Minimum Delay	Selectable Setup	Enable Disable	If enabled, disables TCP Nagle algorithm and sets socket priority to highest Insert check to enable.

Extended Options Window Definition

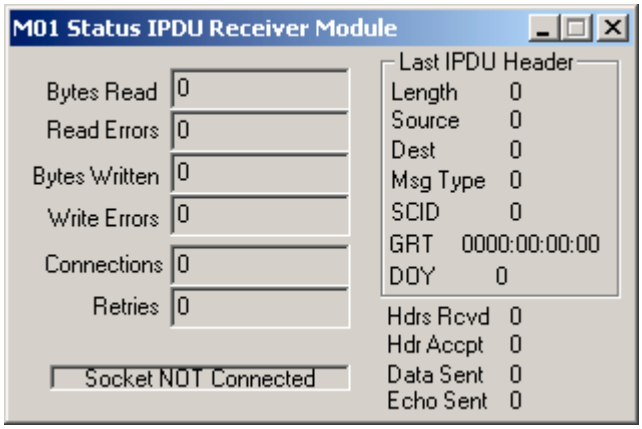
There is no extended options window for the IPDU RECEIVER MODULE.

Configuration Notes

There are no configuration notes for the IPDU RECEIVER MODULE.

Status Window Definition

When you click the Status button on the IPDU RECEIVER MODULE window, a Status IPDU RECEIVER window appears, as shown below.



Module 48-4: Status IPDU RECEIVER MODULE Window

Module 48-5: IPDU RECEIVER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing count	Count of bytes read
Read Errors	Errors	Incrementing count	Count of read errors
Bytes Written	Bytes	Incrementing count	Count of bytes written
Write Errors	Errors	Incrementing count	Count of write errors
Connections	Connections	Incrementing count	Count of connections
Retries	Retries	Incrementing count	Count of retried connections
LAST IPDU HEADER--INFORMATION ABOUT THE LAST RECOGNIZED IPDU PACKET BEFORE ANY FILTERING IS APPLIED			
Length	Decimal	Header value 4-byte field	Last received header data length read from the IPDU header
Source	Decimal	Header value 8-bit field	Last received source code read from the IPDU header
Dest	Decimal	Header value 8-bit field	Last received destination code read from the IPDU header
Msg Type	Decimal	Header value 8-bit field	Last received message type read from the IPDU header
SCID	Decimal	Header value 8-bit field	Last received Spacecraft ID read from the IPDU header
GRT	Decimal	Header value 7-byte field	PB-5 format read from the IPDU header
DOY	Decimal	1 to 366	Calculated from header
GENERAL STATUS			
Hdrs Rcvd	Headers	Incrementing count	Count of all IPDU headers received before any filtering is applied
Hdrs Accpt	Headers	Incrementing count	Count of headers accepted-- number of packets that passed the filtering
Data Sent	Data unit	Incrementing count	Count of data units sent to port 1/uplink path
Echo Sent	Echoes	Incrementing count	Count of echoes sent to port 2/echo path

Module 49 ITCU COMMAND MODULE

MODULE NAME: ITCUCommand.dll

Key Functions

- Provides ITCU200 command interface
- Provides socket connection for ITCU command input and throughputs commands with ITCU header stripped

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the system limitation of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from socket.

CONNECTIONS: Output is to port one.

EVENTS: Neither generates an event nor requires an event input.

Module Functional Description

The ITCU COMMAND MODULE is one of two modules developed to emulate the command and telemetry interface of the RAYTHEON ITCU200 system. The ITCU COMMAND MODULE is a hybrid socket module that registers a single command stream based on the ITCU 200 command interface requirements.

The ITCU COMMAND MODULE receives command data through a socket interface. The module accepts Command Registration requests and provides Command Registration Responses or Errors Responses. Once a command client connection is registered, the module accepts Spacecraft command input. A Command Response is generated, time stamped, and output as soon as the command data is output.

For command input (from socket), the Clients send requests and IDs, the module accepts connections, receives commands, and outputs commands to port one. The module removes ITCU header and verifies source and destination before output to port one.

Input Data Format

Command data with ITCU header.

Output Data Format

Command output with ITCU header removed.

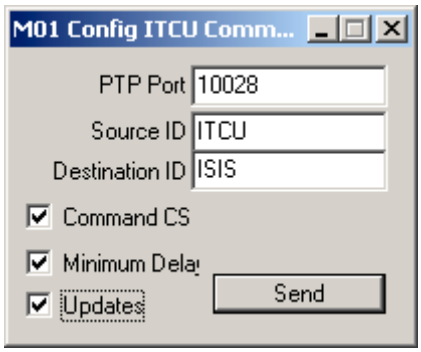
Module Window

When *ITCUCommand.dll* is added to a desktop, the ITCU Command dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 49-1: ITCU COMMAND MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure ITCU COMMAND MODULE Board Parameters sub-menu.



Module 49-2: Config ITCU COMMAND MODULE Window

Module 49-3: ITCU COMMAND MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
PTP Port	Port number		Socket port to be used for TCP transfers.
Source ID	characters		The expected source ID- four characters
Destination ID	characters		The expected destination ID- four characters
Command CS	Selectable Setup	Enable Disable	When enabled (check inserted), it verifies checksum. If an error occurs, it is reported to the status window, and data is still forwarded.
Minimum Delay	Selectable Setup	Enable Disable	When enabled (check inserted), it disables the TCP Nagle algorithm and sets socket priority to highest.

Extended Options Window Definition

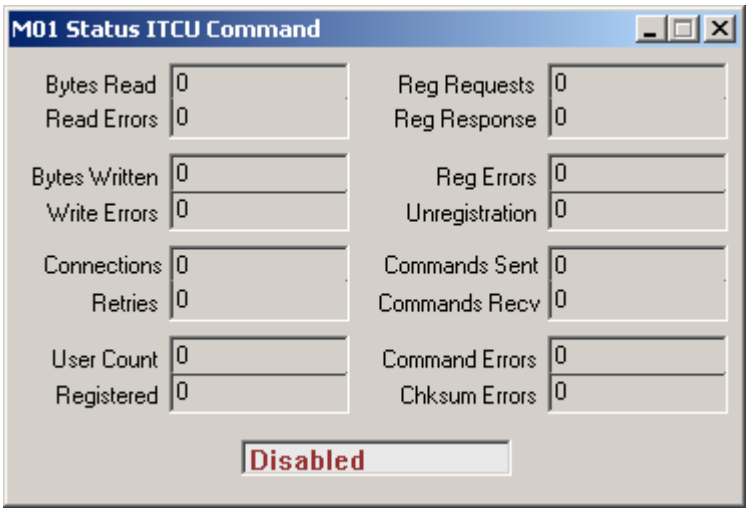
There is no extended options window for the ITCU COMMAND MODULE.

Configuration Notes

There are no configuration notes for the ITCU COMMAND MODULE.

Status Window Definition

When you click the Status button on the ITCU COMMAND MODULE window, a Status ITCU COMMAND window appears, as shown below.



Module 49-4: Status ITCU COMMAND MODULE Window

Module 49-5: ITCU COMMAND MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing count	Count of bytes read.
Read Errors	Errors	Incrementing count	Count of read errors.
Bytes Written	Bytes	Incrementing count	Count of bytes written.
Write Errors	Errors	Count	Count of write errors.
Connections	Connections	Count	Number of connections.
Retries	Retries	Count	Number of connection retries.
User Count	Clients	Count	Number of clients connected to the server.
Registered	Clients	Count	Number of registered clients.
Reg Requests	Requests	Incrementing count	Count of registration requests.
Reg Response	Responses	Incrementing count	Count of registration responses.
Reg Errors	Errors	Incrementing count	Count of registration errors.
Un-registration		Incrementing count	Count of un-registration.

Parameter	Unit	Range or Values	Description
Commands Sent	commands	Incrementing count	Count of commands sent.
Commands Recv	commands	Incrementing count	Count of commands received.
Command Errors	Errors	Incrementing count	Count of command errors.
Chksum Errors	Errors	Incrementing count	Count of checksum errors.
Status window	text	Enabled Disabled	Enabled/Disabled status

Module 50 ITCU TELEMETRY MODULE

MODULE NAME: ITCUTelemetry.dll

Key Functions

- Emulates the ITCU200 telemetry interface
- Outputs data with a header appended to a socket

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the system 32 modules per desktop limitation.

Required Input Source, Connections, and Events

INPUT SOURCE: Data input is expected from the Serial Input Module.

CONNECTIONS: Output is to the socket.

EVENTS: TBS

Module Functional Description

The ITCU TELEMETRY MODULE is one of two modules developed to emulate the command and telemetry interface of the RAYTHEON ITCU200 system. The ITCU TELEMETRY MODULE is a hybrid socket module that provides a socket connection for telemetry stream output based on the ITCU200 telemetry interface.

It is a multi-client server module: up to 31 connections are allowed. Once a telemetry stream is registered with the module, ASCII telemetry data messages are sent to the registered client via the socket interface. The module accepts Telemetry Registration messages and provides a Telemetry Registration response or a Telemetry Registration Error Response. The module un-registers a connection upon receiving a valid Telemetry Un-registration request from a client.

The ITCU TELEMETRY MODULE receives 512 byte telemetry frames from the Serial Input Module and then formats the frame into the telemetry data update message format by pre-pending a header. Header information includes information such as a time stamp field (time stamp from Serial Input Module), frame sync state, bit rate, and checksum.

To use the Status Frames option (see configuration parameters), the Serial Input Module must precede the ITCU TELEMETRY MODULE in the data flow and it must be set so that it generates flywheel frames when it goes out of lock. This means that the Serial Input Module flywheel count must be set to a value greater than zero. The combination of setting the Serial Input

Module to generate flywheel frames, and enabling the Status Frames option on the ITCU TELEMETRY MODULE generates unsolicited telemetry status frames when the synchronizer is out of lock.

Input Data Format

512-byte input from Serial Input Module

Output Data Format

512-byte frame with header prepended.

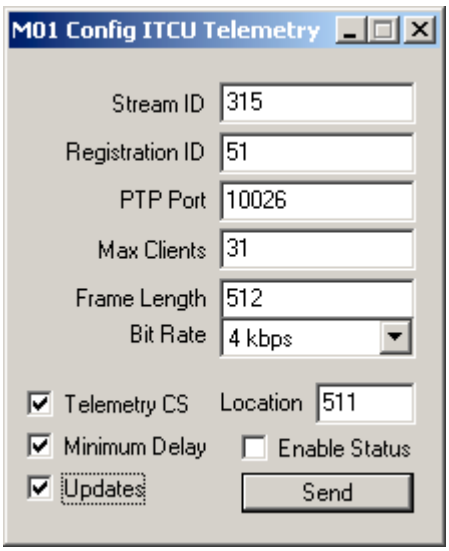
Module Window

When *ITCUTelemetry.dll* is added to a desktop, the ITCU Telemetry dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 50-1: ITCU TELEMETRY MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure ITCU TELEMETRY MODULE Board Parameters sub-menu.



Module 50-2: Config ITCU TELEMETRY MODULE Window

Module 50-3: ITCU TELEMETRY MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Stream ID			
Registration ID			
PTP Port			User selectable socket port to be used for TCP transfers
Max Clients			Count of allowable registered clients
Frame Length			Expected frame length
Bit Rate	Selectable setup	500 bps 1 Kbps 2 Kbps 4 Kbps	Expected bit rate
Telemetry CS		Enable Disable	Enables (check inserted) telemetry checksum
Location			Offset of checksum
Minimum Delay		Enable Disable	Disables (check inserted) TCP Nagle algorithm and sets socket priority to highest
Enable Status		Enable Disable	When enabled (check inserted), enables unsolicited telemetry status frames

Extended Options Window Definition

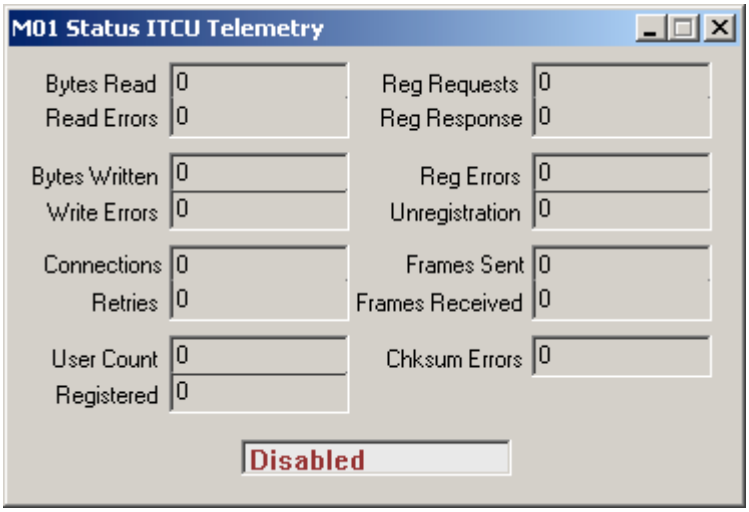
There is no extended options window for the ITCU TELEMETRY MODULE.

Configuration Notes

There are no configuration notes for the ITCU TELEMETRY MODULE.

Status Window Definition

When you click the Status button on the ITCU TELEMETRY MODULE window, a Status ITCU TELEMETRY window appears, as shown below.



Module 50-4: Status ITCU TELEMETRY MODULE Window

Module 50-5: ITCU TELEMETRY MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing Count	Count of bytes read
Read Errors	Errors	Incrementing Count	Count of read errors
Bytes Written	Bytes	Incrementing Count	Count of bytes written
Write Errors	Errors	Incrementing Count	Count of write errors
Connections	Connections	Count	Number of connections
Retries	Retries	Count	Number of connection retries
User Count	Clients	Count	Number of clients connected to server
Registered	Clients	Count	Number of registered clients
Reg Requests	Requests	Incrementing Count	Count of registration requests
Reg Response	Responses	Incrementing Count	Count of registration responses
Reg Errors	Errors	Incrementing Count	Count of registration errors
Unregistration		Incrementing Count	Count of un-registration
Frames Sent	Frames	Incrementing Count	Count of frames sent
Frames Received	Frames	Incrementing Count	Count of frames received
Chksum Errors	Errors	Incrementing Count	Count of checksum errors
Status window	Text	Enabled Disabled	Module enabled/disabled status

Module 51 LEO-T CDH FORMATTER MODULE

MODULE NAME: LeoT CDHFormatter.dll

Key Functions

- Outputs data in LEO-T CDH format (24-byte header appended to frame)
- Provides user-configurable header fields that include source code, destination code, message type, SCID, and software version

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the limitation of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Expects CCSDS frames, but accepts any data input. Expects a time stamp buffer descriptor.

CONNECTIONS: Outputs data to port one, which must be connected to another modules input or the data is lost.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

The LEO-T COMMAND DELIVERY HEADER (CDH) FORMATTER MODULE encapsulates command data for transmission across the network. The LEO-T CDH Formatter accepts data buffers from other modules and prepends the CDH header. The LEO-T CDH Formatter outputs the CDH packet (header and data).

The LEO-T CDH Formatter typically receives command frames with quality annotation from the Serial Input Module or the VCP Module. The LEO-T CDH Formatter is typically connected to a Socket Module for network output.

See **Appendix B** for a detailed overview of the LEO-T CDH format.

Input Data Format

Expects CCSDS frame input, but can processes any fixed buffer input. If status is missing from the buffer descriptor associated with the input, then some of the header information can not be filled.

Output Data Format

Outputs the input buffer (frame) with the LEO-T CDH appended.

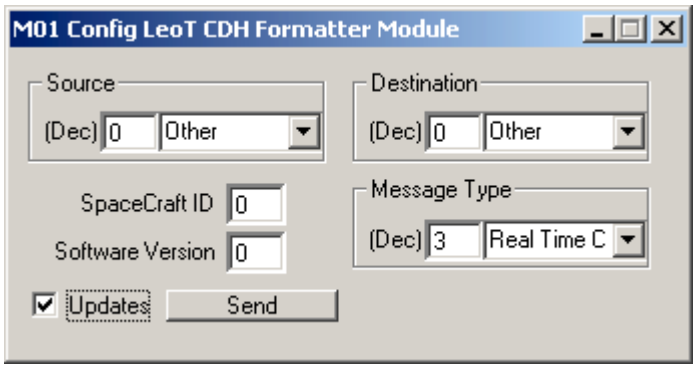
Module Window

When *LeoT CDHFormatter.dll* is added to a desktop, the LEO-T CDH Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 51-1: LEO-T CDH FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure LEO-T CDH FORMATTER MODULE Board Parameters sub-menu.



Module 51-2: Config LEO-T CDH FORMATTER MODULE Window

Module 51-3: LEO-T CDH FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
All configuration fields set up the LEO-T CDH. The values defined here are stamped into the header of each LEO-T CDH packet output.			
SOURCE			

Parameter	Unit	Range or Values	Description
Dec	Decimal	8-bit field	Defines the source code stamped into each header. Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit source code in this window that is stamped into each LEO-T CDH. For any other choice than "other," this window provides the decimal number associated with the selected source. The pull down menu provides the text representation of the most common source codes. When the source is selected and the Send is issued, this window automatically shows the numeric representation for that source-- this is the number that is stamped into each LEO-T CDH.
Pull down menu	Selectable Setup	L7M, SFS, ASK, SPZ, WSC, WIL, VLF, VAF, WFF, or Other	Defines the source code stamped into each header. Provides options for the source code that is stamped into each LEO-T CDH. The operator may select the text representation of a source code, or choose to define a source code by selecting other. Use the text window (Dec) for other definitions. If a text source code is selected, then the numeric representation of the selected source shows in the text window when the SEND is issued.
DESTINATION			
Dec	Decimal	8-bit field	Allows an entry when "other" is selected from the pull-down menu. If other is selected, the operator may define an 8-bit destination code in this window that is stamped into each LEO-T CDH. For any other choice than "other," this window provides the decimal number associated with the selected destination. The pull down menu provides the text representation of the most common destination codes. When the destination is selected and the Send is issued, this window automatically shows the numeric representation for that destination -- this is the number that is stamped into each LEO-T CDH.
Pull down menu	Selectable Setup	L7M, SFS, ASK, SPZ, WSC, WIL, VLF, VAF, WFF, or Other	Provides options for the destination code that is stamped into each LEO-T CDH. The operator may select the text representation of a destination code, or choose to define a destination code by selecting other. Use the text window (Dec) for other definitions. If a text destination code is selected, then the numeric representation of the selected destination shows in the text window when the SEND is issued.
MESSAGE TYPE			

Parameter	Unit	Range or Values	Description
Dec	Decimal	8-bit field	Defines the Message Type that is stamped into each LEO-T CDH. Allows an entry when "other" is selected from the pull down menu. If other is selected, the operator may define an 8-bit message type in this window that is stamped into each LEO-T CDH. For any other choice than "other," this window provides the decimal number associated with the selected message type. The pull down menu provides the text representation of the most common message types. When the message type is selected and the Send is issued, this window automatically shows the numeric representation for that message type -- this is the number that is stamped into each LEO-T CDH.
Pull down menu		Real Time Telemetry, Playback Telemetry, Real Time Command, Command Echo, Other	Defines the Message Type that is stamped into each LEO-T CDH. Provides options for the message type that is stamped into each LEO-T CDH. The operator may select the text representation of a message type, or choose to define a message type by selecting "other." Use the text window (Dec) for other definitions. If a text message type is selected, then the numeric representation of the selected message type shows in the text window when the SEND is issued.
OTHER SETUP			
Spacecraft ID	Decimal	2 byte field	Defines the SCID that is stamped into each LEO-T CDH.
Software Version	Decimal	2 byte field	Defines the EDOS Software version number that is stamped into LEO-T CDH.

Extended Options Window Definition

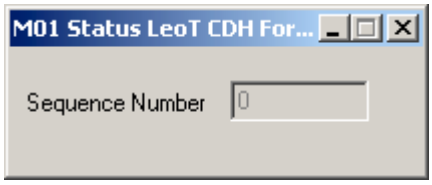
There is no extended options window for the LEO-T CDH FORMATTER MODULE.

Configuration Notes

There are no configuration notes for the LEO-T CDH FORMATTER MODULE.

Status Window Definition

When you click the Status button on the LEO-T CDH FORMATTER MODULE window, a Status LEO-T CDH FORMATTER window appears, as shown below.



Module 51-4: Status LEO-T CDH FORMATTER MODULE Window

Module 51-5: LEO-T CDH FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Sequence Number	integer	2 byte field	Reports the sequence number of the last frame output: sequence number is a field in the LEO-T CDH. Will role over at counter end. Note that this value actually equals output + 1

Module 52 LEO-T CDH RECEIVER MODULE

MODULE NAME: LeoT CDHReceiver.dll

Key Functions

- Accepts Leo-T CDH formatted packets from a socket
- Outputs data to port one for uplink (optionally strips Leo-T CDH)
- Optionally outputs echoes to port 2 and optionally swaps source and destination codes
- Filters all received packets by destination code
- Optionally filters input packets by data type: only type 3, which is forward commands, are output to port one
- Accommodates busy networks by optionally disabling nagle algorithm

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the limit of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Data is input from a socket.

CONNECTIONS: Port 1 output is the data path. Port 2 output is the command echo.

EVENTS: Neither generates an event or expects an event input.

Module Functional Description

The LEO-T CDH RECEIVER MODULE is a special type of Socket Module. It receives LeoT CDH packets from a network socket connection. The LEO-T CDH Receiver filters the packet based on the LEO-T CDH. The LEO-T CDH RECEIVER MODULE has two data output ports. Port 1 outputs accepted packets with or without the LEO-T CDH header. Port 2 is used for command echo and outputs accepted packets with the source and destination code swapped for transmission back to the control center.

Note: See Appendix B for a detailed overview of the LEO-T CDH format.

Input Data Format

Expects LEO-T CDH packet input, which is a frame with the 24 byte LEO-T CDH appended.

Output Data Format

Output to port one is the frame that may or may not include the LEO-T CDH.

Output to port two is either the exact data throughput or it is an echo, which is the input packet with source and destination swapped.

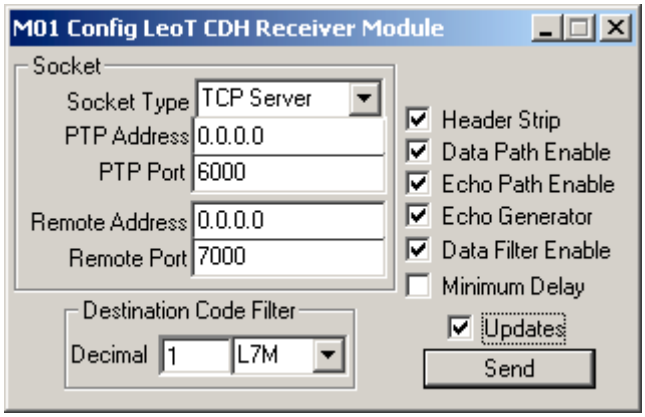
Module Window

When LeoT CDHReceiver.dll is added to a desktop, the LEO-T CDH Receiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 52-1: LEO-T CDH RECEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure LEO-T CDH RECEIVER MODULE Board Parameters sub-menu.



Module 52-2: Config LEO-T CDH RECEIVER MODULE Window

Module 52-3: LEO-T CDH RECEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
SOCKET			
Socket Type	Selectable Setup	Disabled, TCP Server, TCP Client	Sets the socket type: Disabled, UDP, TCP Client, or TCP Server

Parameter	Unit	Range or Values	Description
PTP Address	Address		Local Avtec PTP for Windows address. Address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	Port	1 to 65,535 some are reserved	The local socket that is to be opened: socket number.
Remote Address	Address		Remote IP address. Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal.
Remote Port	port	1 to 65,535 some are reserved	Remote Socket number -- Defines the socket on the remote machine on which it expects data transfer.
DESTINATION CODE FILTER			
Decimal	decimal	8-bit field	Defines the destination code filter: the input header field must match this value or it is filtered. This field accepts a data entry if "other" is chosen from the pull down menu. Otherwise, it reflects the numeric value of the destination code chosen from the pull-down menu. The numeric value appears when SEND is issued.
Pull down menu	Selectable Setup	Disabled L7M, SFS, ASK, SPZ, WSC, WIL, VLF, VAF, WFF, or Other	Defines the destination code filter: the input header field must match the chosen destination code. This is the text representation of the destination code. The numeric representation (the actual header value) shows in the "Decimal" window. The operator may select the text representation of a source code, or choose to define a source code by selecting "other." Use the text window (Dec) for other definitions. If a text source code is selected, then the numeric representation of the selected source shows in the text window when the SEND is issued.
OTHER SETUP			
Header Strip	Selectable Setup	Enable Disable	Defines the data format output on port one. Insert check to enable. Enable – strips header and passes data Disable – passes header and data
Data Path Enable	Selectable Setup	Enable Disable	Allows the operator to turn off the data path (don't send an uplink), which is port 1. Insert a check to enable. If enabled, data is output port 1 (data path) If disabled, no data is output port 1.

Parameter	Unit	Range or Values	Description
Echo Path Enable	Selectable Setup	Enable Disable	Determines if data is output on the echo path (port 2). If enabled, data is always output on port 2. If the echo generator is also enabled, then each received LEO-T CDH packet is output in echo format (source and destination swapped). If the echo generator is not enabled, data is throughput with no changes. If disabled, no data is output on port 2. Insert check to enable.
Echo Generator	Selectable Setup	Enable Disable	When echo path is enabled, this switch determines if data is output as an echo. Insert check to enable. If enabled (and echo path enabled), outputs each forward command as an echo– swaps destination and source code in header and sends command echo. If disabled (and echo path enabled), then data output port 2 as it is received (no header changes).
Data Filter Enable	Selectable Setup	Enable Disable	Determines if the data type field is checked and data filtered according to the data field value. Allows the operator to output only commands on data port 1, which is the uplink path. (If test bit steering is enabled, that logic is applied before data filter; therefore, the data filter logic is never applied). When enabled, works as a Command filter-- determines if it is an uplink/forward command and passes only commands out Data Port 1. By default, the filter is message type 3. So if enabled, the only thing accepted into uplink is type 3, which is a forward command If disabled, does not filter on data type field. Should disable if module is being used for telemetry.
Minimum Delay	Selectable Setup	Enable Disable	If enabled, disables TCP nagle algorithm and sets socket priority to highest Insert check to enable.

Extended Options Window Definition

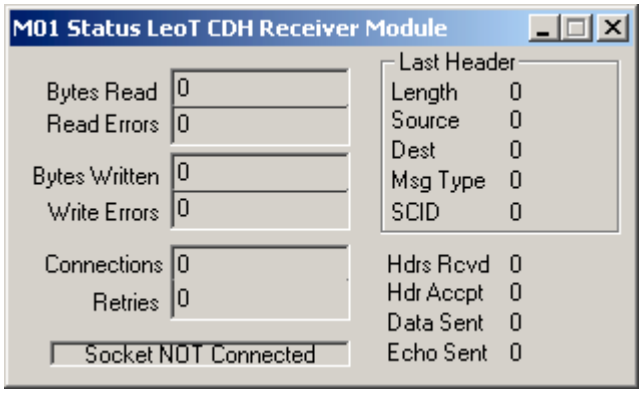
There is no extended options window for the LEO-T CDH RECEIVER MODULE.

Configuration Notes

There are no configuration notes for the LEO-T CDH RECEIVER MODULE.

Status Window Definition

When you click the Status button on the LEO-T CDH RECEIVER MODULE window, a Status LEO-T CDH RECEIVER window appears, as shown below.



Module 52-4: Status LEO-T CDH RECEIVER MODULE Window

Module 52-5: LEO-T CDH RECEIVER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing count	Count of bytes read
Read Errors	Errors	Incrementing count	Count of read errors
Bytes Written	Bytes	Incrementing count	Count of bytes written
Write Errors	Errors	Incrementing count	Count of write errors
Connections	Connections	Incrementing count	Count of connections
Retries	retries	Incrementing count	Count of retried connections
LAST HEADER--INFORMATION ABOUT THE LAST RECOGNIZED PACKET BEFORE ANY FILTERING IS APPLIED			
Length	Decimal	Header value 2-byte field	Last received header data length read from the LEO-T CDH
Source	Decimal	Header value 8-bit field	Last received source code read from the LEO-T CDH
Dest	Decimal	Header value 8-bit field	Last received destination code read from the LEO-T CDH
Msg Type	Decimal	Header value 8-bit field	Last received message type read from the LEO-T CDH
SCID	Decimal	Header value 2-byte field	Last received Spacecraft ID read from the LEO-T CDH
OTHER STATUS			
Hdrs Rcvd	Headers	Incrementing count	Count of all LEO-T CDH headers received before any filtering is applied
Hdr Accpt	Headers	Incrementing count	Count of headers accepted-- number of packets that passed the filtering

Parameter	Unit	Range or Values	Description
Data Sent	Data Units	Incrementing count	Count of data units sent to port 1/uplink path
Echo Sent	Echoes	Incrementing count	Count of echoes sent to port 2/echo path

Module 53 LEO-T TFDH RECEIVER MODULE

MODULE NAME: LeoT_TFDHReceiver.dll

Key Functions

- Accepts frames with LEO-T TFDH appended to each from a socket
- Provides UDP, TCP Server, TCP Client, and Multicast socket connections
- Support multi-homed machines
- Optionally strips LEO-T TFDH from each frame, and then outputs frames to port one
- Supports high-latency networks
- Provides TCP keep alive capability

Required Hardware

- No special hardware is required.

Add Module Guidelines

There are no limitations to the number of LEO-T TFDH RECEIVER MODULES that can be added to a desktop as long as the system limit of 32 modules per desktop is not exceeded.

Required Input Source, Connections, and Events

INPUT SOURCE: Data is input from a socket connection.

CONNECTIONS: Data is output to port one: data is lost unless the output is connected to another module's input port.

EVENTS: This module neither generates an event nor requires an event input.

Module Functional Description

The LEO-T TFDH RECEIVER MODULE connects to a socket, receives frames with LEO-T TFDH appended, and then throughputs the frames to port one output. The header can be removed or throughput with each frame.

To protect against a broken pipe, if the module reads a frame length from the header is greater than 16Kbytes, it assumes that the data is skewed and disconnects from the socket. It throws away the bad frame and waits for the source to reconnect. This allows the source to recognize that data is being corrupted (when the disconnect is received) and then reinitialize and start with a new frame.

The module also optionally provides keep alive and minimum delay functions (minimum delay can be used for high latency networks).

Refer to **Appendix B** for a definition of the LEO-T TFDH. It is a 20 byte header that includes the following information: header version number; length of frame in bytes including header; status flags; and ERT in PB-5 format.

Input Data Format

Expects frames with the 20 byte TFDH prepended.

Output Data Format

There are two possible output formats:

1) Frames with TFDH header removed
2) Frames with TFDH still appended (data is throughput)

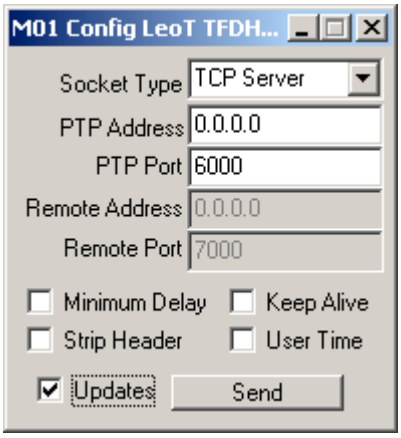
Module Window

When *LeoT_TFDHReceiver.dll* is added to a desktop, the LEO-T TFDH Receiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 53-1: LEO-T TFDH RECEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure LEO-T TFDH RECEIVER MODULE Board Parameters sub-menu.



Module 53-2: Config LEO-T TFDH RECEIVER MODULE Window

Module 53-3: LEO-T TFDH RECEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Socket Type	Selectable Setup	UDP TCP Client TCP Server Multicast	Choice of UDP, TCP Server, TCP Client, or Multicast If set to server, it opens up the port and waits for connection. If set to client, connects to a server that has already opened port. In client mode, it keeps trying to connect.
PTP Address	address	N/A	Local Avtec PTP for Windows address. Address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	port	1 to 65,535 some are reserved	The local socket that is to be opened. Socket number (use for TCP Server or UDP Receiver)
Remote Address	address	N/A	Remote IP address (only used for TCP client, Multicast or UDP transmit) Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal. Applies to UDP, TCP Client, and Multicast Socket Types.
Remote Port	port	1 to 65,535 some are reserved	Remote Socket number (only used for TCP client, Multicast, or UDP transmit) Defines the socket on the remote machine on which it expects data transfer
Minimum Delay	Selectable Setup	Enable Disable	Applies only to TCP modes. When enabled, the protocol stack buffer size is 512 kB. It also disables the Nagle algorithm. Designed for high latency networks.
Strip Header	Selectable Setup	Enable Disable	If enabled, strips the TFDH header from each frame received on the socket before output to port one. If disabled, each frame is output with the header still attached (10 byte header) Insert check to enable.
Keep Alive	Selectable Setup	Enable Disable	Applies to TCP only. Network integrity test. Uses an inactivity timer that starts when no data has been sent/received on the connection. Used to decide if the connection is broken. Works like a ping. It sends an Out of Band message on the socket and expects a return message also on the Out of Band. If Keep Alive message is not answered: client tries to reconnect and server disconnects everyone and then waits for a reconnect.

Parameter	Unit	Range or Values	Description
User Time	Selectable Setup	Enable Disable	When enabled, the Avtec PTP for Windows buffer descriptor associated with the input (from socket) frame is stamped with the time read from the frame's LEO-T header. --In most cases, the header time reflects the original GRT of the frame. By putting the ERT in the buffer descriptor, ERT is available to other modules downstream. Insert check to enable.

Extended Options Window Definition

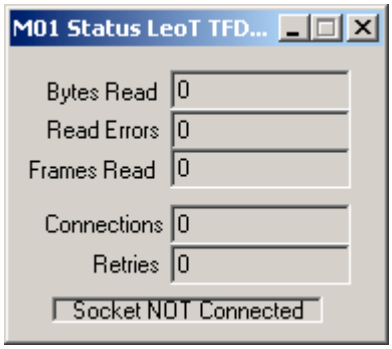
There is no extended options window for the LEO-T TFDH RECEIVER MODULE.

Configuration Notes

There are no configuration notes for the LEO-T TFDH RECEIVER MODULE.

Status Window Definition

When you click the Status button on the LEO-T TFDH RECEIVER MODULE window, a Status LEO-T TFDH RECEIVER window appears, as shown below.



Module 53-4: Status LEO-T TFDH RECEIVER MODULE Window

Module 53-5: LEO-T TFDH RECEIVER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing counter	Total bytes received on the socket. Count of bytes read
Read Errors	Errors	Incrementing counter	Count of read errors Read error = socket failure--a request to read bytes from the host is meet with a network error. Module re-initializes when a read error is received.
Frames Read	Frames	Incrementing counter	Number of frames received on the socket

Parameter	Unit	Range or Values	Description
Connections	Connections	Incrementing counter	Current number of connections. Number of times a successful connection has been made includes re-connections.
Retries	retries	Incrementing counter	Number of connection retries-- number of times a connection was attempted
Status Window	Text status	Socket not connected, Connected	Reports state of the socket

Module 54 LEO-T TFDH TRANSMITTER MODULE

MODULE NAME: LeoT TFDHTransmitter.dll

Key Functions

- Outputs LEO-T TFDH packets to a socket connection
- Appends LEO-T TFDH to input frames
- Provides user-configurable header fields

Required Hardware

- No special hardware is required.

Add Module Guidelines

Do not exceed the 32 modules per desktop limitation. No other LEO-T TFDH TRANSMITTER MODULE limitations are imposed.

Required Input Source, Connections, and Events

INPUT SOURCE: Input is from another module. Typical source modules are the Serial Input and VC Processor modules.

CONNECTIONS: Outputs data to a socket. Outputs the same data sent to socket on ports 1 and 2.

EVENTS: Neither generates an event nor requires an event input.

Module Functional Description

The LEO-T TFDH (Telemetry Frame Delivery Header) TRANSMITTER MODULE encapsulates data for transmission across the network. The LEO-T TFDH Transmitter accepts data buffers from other modules and prepends the TFDH header to the data. The LEO-T TFDH Transmitter outputs the TFDH packet (header and data).

Typically, data input is telemetry frames with quality annotation from the Serial Input Module or the VCP Module. The ERT stored in the Avtec PTP for Windows buffer descriptor is used to time stamp the TFDH.

See **Appendix B** for a detailed overview of the LEO-T TFDH format. It is a 20-byte header that includes the following information: header version number; length of frame in bytes including header; status flags; and ERT in PB-5 format.

Input Data Format

The module expects frame input that includes status (buffer descriptor information), which means the frame was processed by either or both the Serial Input and VCP modules. The module accepts any fixed-length buffer input, but unless it is frame format with status, the output header is incomplete.

Output Data Format

Output is the input buffer (frame) with the LEO-T TFDH appended.

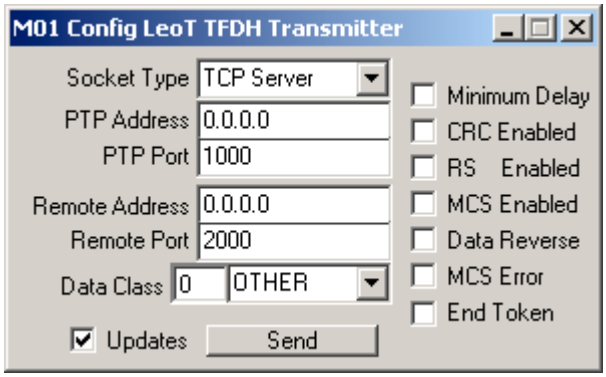
Module Window

When *LeoT TFDHTransmitter.dll* is added to a desktop, the LEO-T TFDH Transmitter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 54-1: LEO-T TFDH TRANSMITTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure LEO-T TFDH TRANSMITTER MODULE Board Parameters sub-menu.



Module 54-2: Config LEO-T TFDH TRANSMITTER MODULE Window

Module 54-3: LEO-T TFDH TRANSMITTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Socket Type	Selectable Setup	Disabled TCP Server TCP Client	Sets the socket type
PTP Address	Address	IP Address	Local Avtec PTP for Windows address. Address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	Port	1 to 65,535 some are reserved	The local socket that is to be opened. Enter the socket number
Remote Address	Address	IP Address	Remote IP address. Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal.
Remote Port	Port	1 to 65,535 some are reserved	Remote Socket number Defines the socket on the remote machine on which it expects data transfer
TOGGLES & HEADER FIELD DEFINITIONS--most of configuration in this section defines fields in the LEO-T TFDH, but Minimum Delay and End Token are exceptions. Both of these switches toggle functionality. For the other configuration, the selections made determine the values that are filled into each TFDH that is appended to frames prior to socket output.			
Data Class	Decimal and text	OTHER (5 bits) CCSDS Frames, CCSDS Packets, TDM, TDM Stripper	Defines the Data Class header field that is stamped into each LEO-T TFDH packet output. Two fields define the data class: pull-down text selection and numeric entry field. The pull down menu lists the standard data types. When a selection is made from the pull down and then the SEND issued, the numeric field automatically fills in with the value that is stamped into the header. 1 = CCSDS Frame, 2 = CCSDS Packet, 3 = TDM Frame, 4 = Stripped TDM Frame To make a manual entry that does not appear in the standard selection list, choose "other" from the pull-down. When "other" is selected, the operator can enter a value into the entry field, and then that value is stamped into each header.
Minimum Delay	Selectable Setup	Enable Disable	When enabled, the module disables TCP nagle algorithm and sets socket priority to highest. Insert check to enable.
CRC Enabled	Selectable Setup	Enable Disable	**Sets the bit in the header** Enabled – module adds CRC state to header
RS Enabled	Selectable Setup	Enable Disable	**Sets the bit in the header** Enabled – module adds Reed-Solomon state to header

Parameter	Unit	Range or Values	Description
MCS Enabled	Selectable Setup	Enable Disable	**Sets the bit in the header** Enabled – module adds Master Channel Sequence check state to header
Data Reverse	Selectable Setup	Enable Disable	Controls the setting of the data forward/reverse flag in the header. If enabled (insert check), the flag is set in the header (indicates reverse data). If disabled, the flag is NOT set in the header (indicates forward data)
MCS Error	Selectable Setup	Enable Disable	**Sets the bit in the header** Enable – MCS error number increases by 2 or more, Disable – number increases monotonically.
End Token	Selectable Setup	Enable Disable	Controls how the socket is closed If enabled (insert check), then prior to closing socket, module will send a 10 byte SMEX header without frame to destination. If disabled, no close message is sent.

Extended Options Window Definition

There is no extended options window for the LEO-T TFDH TRANSMITTER MODULE.

Configuration Notes

There are no configuration notes for the LEO-T TFDH TRANSMITTER MODULE.

Status Window Definition

When you click the Status button on the LEO-T TFDH TRANSMITTER MODULE window, a Status LEO-T TFDH TRANSMITTER window appears, as shown below.

Module 54-4: Status LEO-T TFDH TRANSMITTER MODULE Window

Module 54-5: LEO-T TFDH TRANSMITTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Bytes Written	Bytes	Incrementing count	Count of bytes written Includes the header bytes
Write Errors	Errors	Incrementing count	Count of write errors
Connections	Connections	Incrementing count	Count of connections
Retries	Retries	Incrementing count	Count of retried connections

Module 55 MANUAL COMMAND MODULE

MODULE NAME: ManualCommand.dll

Key Functions

- Generate a buffer containing user specific data and send this data onto another module.

Required Hardware

- None

Add Module Guidelines

There are no limits as to how many iterations of the Manual Command module that may be loaded. One iteration may output user specified data to as many as four (4) different modules, using module connections ports 1 thru 4.

Required Input Source, Connections, and Events

INPUT SOURCE: This module chiefly receives data via user entry. These entries may come from the console gui or from the Remote Interface Library (RIL) user function calls. The user may also specify the name and path of a file (default: \ptp_user\scripts\default.man) containing a user-specified command mnemonic and the actual hex, octal, decimal, binary, or text of the data to be sent when that mnemonic is called up.

CONNECTIONS: Ports 1 thru 4 of this module should be connected to another module requiring an input buffer. Only those ports that the user intends to use need to be connected.

EVENTS: No events are generated or expected.

Module Functional Description

This module sends a buffer of data containing user-specified data. Up to 100 bytes may be sent within one buffer. A buffer length of zero will not be sent. The buffer is time stamped with the time that the user sent the buffer on to another module.

A few different input methods may be used to drive this module. Data may be entered in binary, octal, hexadecimal, decimal or text. Each byte is separated by a space, because ambiguity would result with octal representation. Input from the keyboard is accepted by the console gui. The last 100 buffers sent are retained in a list box, allowing the user to re-send a

buffer or recall a previous buffer and edit the content prior to sending. Data may also be entered into the module using the Remote Interface Library (RIL) function calls.

Text formatted files may be created by the user. These files can contain up to thirty commands which may be sent at will by the user. A name is assigned to each command, so that the user may refer to the content easily. The default file is named "\ptp_user\scripts\default.man". These text files may be created by the Notepad program (included with Windows NT) or any other text editor. The format of each of the lines (up to 30) of the text file is:

Mnemonic=commandcontent

For example:

Xpndr2On=42 37 2C 9B F2 5A

would send six bytes containing 0x42372C9BF25A in a new, time stamped buffer, assuming the user had selected the hexadecimal mode. There are no spaces around the equal sign. No bytes are automatically appended or prepended to the bytes entered by the user. If this capability is desired, the user can feed the data thru the Ames Command Encapsulator or the Stripper module. Predefined commands are sent using the Command window. User entries are sent using the Config window.

Input Data Format

This module expects valid user data in the base selected. For example, only the digits 0 thru 7 are valid in octal. The entire user input is scanned for syntax error before any of the buffer is sent. Any errors in the user input cause the entire buffer not to be sent. This insures only valid data is sent and that the user knows what has or has not been sent at any time.

Output Data Format

The output data is not formatted. No fields are expected or recognized in the input data, as far as the Avtec PTP for Windows is concerned. Only data that was input is sent. This module must be enabled in order to send anything at all.

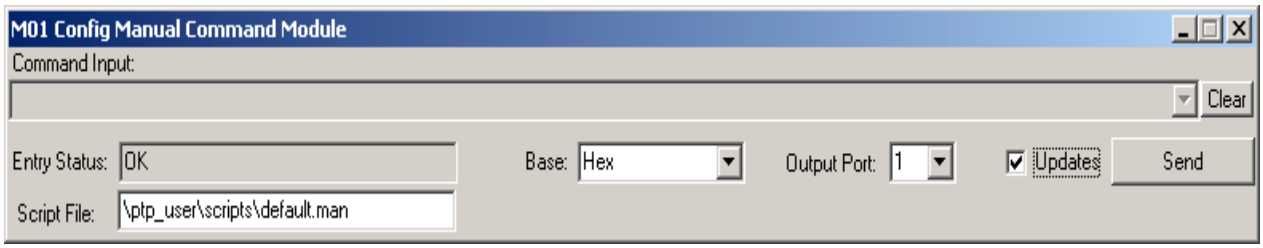
Module Window

When ManualCommand.dll is added to a desktop, the Manual Command dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 55-1: MANUAL COMMAND MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure MANUAL COMMAND MODULE Board Parameters sub-menu.



Module 55-2: Config MANUAL COMMAND MODULE Window

Module 55-3: MANUAL COMMAND MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Command Input	Text, formatted in binary, octal, hexa-decimal, decimal or text	Any valid data for the specified base. See the Base entry.	Command data. Actually, the module does not recognize any part of the data entered so this module may be used to feed ANY type of data to any module.
Entry Status	Display only	Text	Displays the status of the user input. Reports syntax errors or byte count sent status.
Script File	File	Any path and file	Current file selected and available to user to send predefined buffers of data by mnemonic name.
Base	Selection	Hex Decimal Octal Binary Text	Entry base. User input is validated according to this base.
Output Port		1 to 4	Selects the port on which to send the buffer to another module. The port must be connected in order for anything to be sent.
Clear	Button	Push	Clears the input window. Also clears the history in the input list box. The last 100 entries are saved in the list box unless and until the user clears them.

Extended Options Window Definition

There is no extended options window for the MANUAL COMMAND MODULE.

Configuration Notes

When you click the Command button on the MANUAL COMMAND MODULE window, a Command MANUAL COMMAND window appears, as shown below.

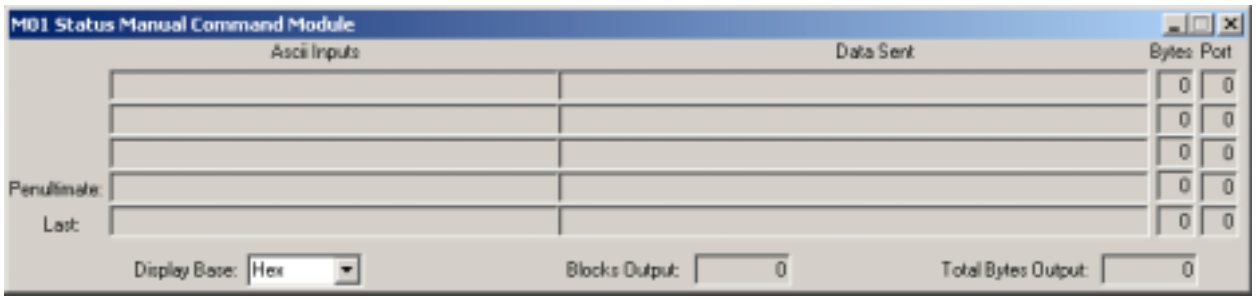
Module 55-4: Command MANUAL COMMAND MODULE Window

Module 55-5: MANUAL COMMAND MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
Command	N/A	No Command Selected, followed by a list of up to 30 user defined command mnemonics	Command data to be sent. The user must ensure that the base of the text in the file agrees with the user-defined setting on the config gui. The user can pick any of the mnemonics and push the Send Command button. The base, and the port parameters are used from the Config window to drive the syntax checking and the output port selection. The results of the send are shown the in Status window. Selecting "No command Selected" causes the Send Command button to be ignored until a command is chosen.
Load List	N/A	Push Button	Re-loads the commands from the user specified command input file. This option is handy if the file has recently been edited by another user.

Status Window Definition

When you click the Status button on the MANUAL COMMAND MODULE window, a Status MANUAL COMMAND window appears, as shown below.



Module 55-6: Status MANUAL COMMAND MODULE Window

Module 55-7: MANUAL COMMAND MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Ascii Inputs	The last 5 User Entries	Any user entry	User input or predefined command input
Data Sent	The last 5 data buffer contents		The first bytes (whatever fits in display) of user input or predefined input.
Bytes	The last 5 buffer byte counts		The bytes count of each of the last five buffers.
Port	The last 5 buffer output port numbers	1 to 4	The port number of the last 5 buffers.

Parameter	Unit	Range or Values	Description
Penultimate	The next to the last buffer sent		This row display the next to the last buffer sent.
Last	The last buffer sent		This row shows the last buffer sent. The window scrolls as each buffer is sent.
Display Base	Base for the Status display ONLY	Hexadecimal Octal Decimal Binary Text	This is the base for the status display window ONLY. This value has no effect on user input. Data may be entered in one base, but shown on the status window in any other.
Blocks Output	Count		Total number of blocks output from the module
Total Bytes Output	Count		Total number of bytes output from the module.

Module 56 NASCOM BLOCKER MODULE

MODULE NAME: NascomBlocker.dll

Key Functions

- Provides NASCOM Block encapsulation of input data stream
- Provides data packing options-- controls how data is packed into block data field
- Allows user-configurable fill pattern
- Optionally disables full block flag
- Optionally calculates BED and places it in the block trailer
- Provides multiple NASCOM Block formats
- Provides user-configurable NASCOM Block header fields
- Time stamps block header
- Optionally, generates Circuit Assurance Blocks
- Allows user control of hold bit status regardless of command or telemetry block type

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of NASCOM BLOCKER MODULES may be loaded in a desktop. Limitations are system imposed. The maximum number of modules allowed in a desktop is 32 and CPU usage should be monitored since this is a software module.

Required Input Source, Connections, and Events

INPUT SOURCE: Input data source may be any module.

CONNECTIONS: This modules input port must be correctly connected to the source module output port. This module outputs to port one. If the module output is not input to another module, the data is lost.

EVENTS: This module neither requires an event input nor generates any events.

Module Functional Description

The NASCOM BLOCKER MODULE is an encapsulation module. It accepts a serial input telemetry data stream and outputs the data stream packaged in NASCOM 4800 bit blocks (4800BB).

The module provides multiple NASCOM Block formats that can be used for data encapsulation. Each format has a unique header and data field length. Using the NASCOM BLOCKER MODULE configuration window, the user can select which Block Type is to be used and provide the information to be placed in the 4800BB header.

Note: NASA Document STDN-502.5 provides a definition of each NASCOM Block type.

Input Data Format

The NASCOM Blocker puts a wrapper around data based on setup regardless of what data is input, so any input data format can be processed. However, input is typically synchronous or asynchronous frame data.

Output Data Format

Output is 600 byte NASCOM Blocks. Each Block has a header, data field, and trailer. The size and contents of the three block components are determined by the Block Type chosen and user-configuration.

When the module is NOT set up for Thruput Mode, there can be fill at the end of the NASCOM Block data field (unless the data unit size is an even multiple of the block data field length). Fill may appear in every block or in a predictable sequence of blocks. If the data unit size is less than the block data field size, then there is fill in every block. If the data unit size spans multiple block data fields, then there is fill in the last block that contains the data unit. The fill pattern is user-configurable. When a data unit requires multiple blocks for encapsulation, a block header flag can optionally be set to indicate the last block that contains a data unit.

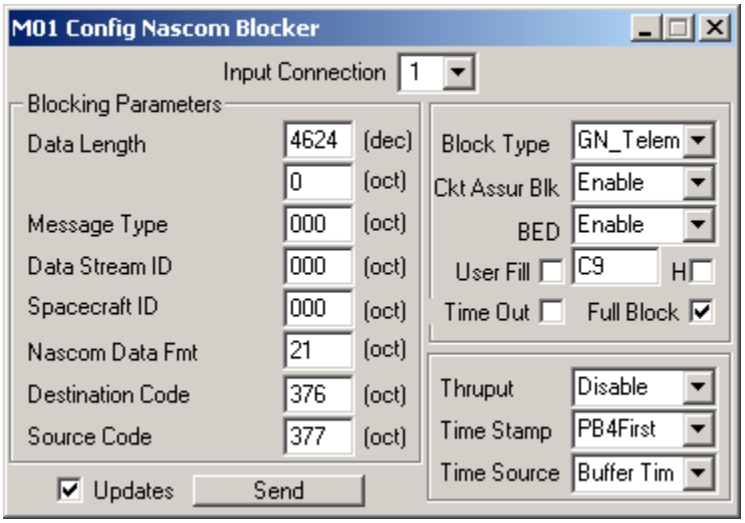
Module Window

When *NascomBlocker.dll* is added to a desktop, the NASCOM Blocker dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 56-1: NASCOM BLOCKER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NASCOM BLOCKER MODULE Board Parameters sub-menu.



Module 56-2: Config NASCOM BLOCKER MODULE Window

Module 56-3: NASCOM BLOCKER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Input Connection	Selectable Setup State	1 to 32	Specifies which Module (by number) is the data stream source **Be certain to set up this field before all other configuration information on the window.** This field must be set to the module number of the input source module. All other setup parameters on this window are not valid until the Input Connection is correctly set. If this field is set after the other configuration information is defined, then the other configuration information is lost when the correct Input Connection is set.
BLOCKING PARAMETERS: The fields in this section define the NASCOM block header fields and the block data field length. The header fields that require definition and the default maximum data field length change for the Block Type that is chosen. **Setting these fields up before the Block Type is chosen may cause information to require re-entry once the Block Type is chosen.** Default values show up for each field and the default value is used for data length or header field definition unless the user enters a new value. Header field values are stamped into each NASCOM block header that is output.			
Data Length	decimal	Varies with block type--must not exceed data field length of block type and must be a multiple of 8	This field defines the length of data to be inserted into 4800BB.It must be set up for all Block Types. When the Block Type is chosen, a default value automatically appears in this field. The default value is the maximum data field length for the selected Block Type. If this value is sets to less than the maximum, then the module packs the defined length of input data into each block and fills the remainder of the field with the fill pattern. The length must be divisible by 8, if it is not the module automatically rounds up and fills the field on an 8-bit boundary. **.**
Message Type	octal	0-377	Defines Message Type field in the block header. Required by GN, DGIB, JSC, Shuttle, and Delta Block Types.

Parameter	Unit	Range or Values	Description
Data Stream ID	octal	0-377	Defines Data Stream Identification header field. Applies to GN, and DGIB Block Types.
Spacecraft ID	octal	0-377	Defines Spacecraft ID header field Applies to GN, DGIB, and Delta Block Types.
NASCOM Data Fmt	octal	0-377	Defines NASCOM Data Format header field. Applies to GN, DGIB, JSC, Shuttle, and Delta Block Types
Destination Code	octal	0-377	Defines to Destination Code header field. Applies to GN, DGIB, JSC, and Delta Block Types.
Source Code	octal	0-377	Defines Source Code header field. Applies to all Block Types.
BLOCK TYPE PARAMETERS			
Block Type	Nascom Block Format	GN_Telem MDM DGIB_Telem JSC Shuttle Delta GN_Comm DGIB_Comm JPL-TCA	Defines which NASCOM Block Type format is to be used. The block type chosen determines which of the Blocking Parameters is applicable. When a Blocking Parameters does not apply to a block format, it is grayed out on the config window. Available Block Types are: 1) GN_Telemetry: NASA Ground Network or DDPS2 or DDPS Thruput format 2) MDM: Johnson Space Center Mux-Demux or Shuttle Thruput format 3) DGIB_Telemetry: DSN to Ground Network Interface Block format 4) JSC: Johnson Space Center Telemetry Block format 5) Shuttle: Space Shuttle format 6) Delta: Modified DDPS Synchronous Blocks for Delta rocket 7) GN_Command: This is the NASA Ground Network command block format. As standard with command block formats it always has a message type of 105 (telemetry is 106), and the NASCOM Header also indicates that it is a command because the NASCOM Data Format byte 5 is an octal 11 (verses telemetry which is 21). Command block format also includes a hold bit, set at bit 66, which is a spare bit in telemetry blocks. Hold bit set to one means don't hold the block and output it immediately 8) DGIB_Command: DSN command block format. All standard command block formats described in 7) also apply to this format. 9) JPL-TCA: similar to DGIB_Telemetry, but specific to JPL. Will not insert CRC and will not allow full block size flag to be set.

Parameter	Unit	Range or Values	Description
Ckt Assur Blk	Selectable Setup State	Enable Disable	When this mode is enabled, the module ensures that one NASCOM Block is output every 1 second regardless of input data. If there is no data to be transmitted, then it generates a Circuit Assurance Block (CAB). A CAB is indicated by a value of zero in the Data Length Field and no data in the data field. When enabled, the module starts counting after each block output, and if one second is reached before another block is ready for output, then a CAB is generated and sent.
Bed	Selectable Setup State	Enable Disable	Block Error Detector – When enabled, the module generates PEP code and appends it to the end of each NASCOM block, i.e, when enabled, the value is calculated and put into the trailer of each NASCOM Block output.
User Fill	hex	Enable Disable 1 byte	This field is controlled by both a check box and an entry field. The check box enables and disables the function: enter a check to enable the function. When enabled, the user defines the fill pattern to be used in the block data field. When enabled, enter the user-defined fill pattern in the entry field.
H	Selectable Setup State	Enable Disable	This field allows the operator to change the "normal" setting of the hold mode bit. Normally, the hold mode bit (66) is set for every command block output and it is not set for every telemetry block output. If this mode is enabled, the opposite condition occurs. If command blocks are output, then the hold mode bit is NOT set. If telemetry blocks are output, then the hold mode bit IS set. To enable, enter a check into the check box
Time Out	Selectable Setup State	Enable Disable	This field defines what is done with a partial block (data field is not full) and no more data is input for one second. When enabled, it waits one second to fill a block, and then if no more data is input, fills the remaining data field with the fill pattern, and then outputs the block. To enable, enter a check into the check box.
Full Block	Selectable Setup State	Enable Disable	If check box is on, then full flag is active and it is set according to the block data field state (if the whole field is user data it is considered full, i.e., there is no fill data in the data field, and the block full flag is set). If check box is off, the full flag is never set.
HEADER TIME STAMP AND DATA FIELD PACKING			

Parameter	Unit	Range or Values	Description
Thruput	Selectable Setup State	Enable Disable	This field defines how data is packed into the NASCOM Block data field. If enabled, the module packs the input data length into the NASCOM Block data field back to back until the block data field is full. Data is not packed according to input data length. When enabled, it ensures that each block's data field is full. NASCOM Blocker will hold block until it is full, and then transmit. If disabled, the Blocker packs the NASCOM Block data field by data unit, which results in fill at the data unit end (depending on data size may be one data unit per block or multiple blocks per data unit);.
Time Stamp		PB4First PB1First PB4Last PB1Last	Defines the time stamp format in the NASCOM Block header. Must have valid time stamp and valid bit rate.
Time Source	Selectable Setup State	Buffer Time PC Time ReCalculate AXAF ReCalc	Defines the Time Source for the time that is stamped into the NASCOM Block header. If set to Buffer Time, module uses the buffer descriptor time. This means the data unit time stamp that is input to the NASCOM Block with the data unit is used in the header of each NASCOM Block that contains that data unit. The origination of this time depends on the data source. Typically, the Serial Input Module precedes the NASCOM Blocker at some point in the data stream and the time tag comes from that module. Depending on the Serial Input Module setup, the time can either be from a time board or it is CPU time. If there is no buffer descriptor time available (buffer time is all zeros), then it will get PC time at data unit input to module. If set to PCTime, then the module stamps the time that the data unit was received by the blocker into the block header. In this mode, it gets "best time," which will be the Apogee time board time if it is installed, or the PC time if the board is not installed. ReCalculate: adjust time in block to always indicate the time when first bit or last bit of data in the block was received. AXAF ReCalc: adjust time to compensate for the inserted AXAF header.

Extended Options Window Definition

There is no extended options window for the NASCOM BLOCKER MODULE.

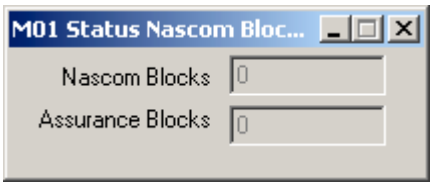
Configuration Notes

To avoid inadvertently configuring this module with the wrong setup, be sure to follow two initial setup steps. First, define the Input Connection (module number of the source data) and

second define the Block Type. Define the other configuration information after these two steps.

Status Window Definition

When you click the Status button on the NASCOM BLOCKER MODULE window, a Status NASCOM BLOCKER window appears, as shown below.



Module 56-4: Status NASCOM BLOCKER MODULE Window

Module 56-5: NASCOM BLOCKER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
NASCOM Blocks	Blocks	Increments by one for each block output	Displays number of NASCOM Blocks transmitted.
Assurance Blocks	Blocks	Increments by one for each block output	If Ckt Assur Blk is enabled, displays number of Assurance Blocks transmitted.

Module 57 NASCOM DEBLOCKER MODULE

MODULE NAME: NascomDeblocker.dll

Key Functions

- Receives NASCOM Block input, deblocks the data, and the outputs the results
- Optionally generates echoes for every block input
- Provides block filtering based on first 12 bytes of each block input
- Optionally provides hold bit control
- Validates input block CRC and optionally filters on CRC errors

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of NASCOM DEBLOCKER MODULES may be loaded in a desktop. Limitations are system imposed. The maximum number of modules allowed in a desktop is 32 and CPU usage should be monitored since this is a software module.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide the NASCOM Block input.

CONNECTIONS: The deblocked data is output on port one. Block echoes are output on port two.

EVENTS: No events are required or generated.

Module Functional Description

The NASCOM Deblocker Module accepts NASCOM 4800BB based on user-defined Block Header Filter and outputs a serial telemetry data stream.

Note: NASA Document STDN-502.5 provides a definition of each NASCOM Block type.

Input Data Format

The module expects NASCOM Block input.

Output Data Format

Output data is either deblocked data or NASCOM Block echoes. For deblocked data, the Block packaging is stripped and the data field is output according to the data length in the block header. For echoed blocks, the block is output with source and destination fields swapped from the input block and with a new polynomial value calculated and put in the trailer.

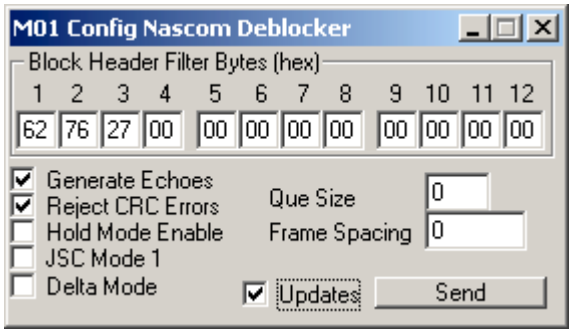
Module Window

When NascomDeblocker.dll is added to a desktop, the NASCOM Deblocker dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 57-1: NASCOM DEBLOCKER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NASCOM DEBLOCKER MODULE Board Parameters sub-menu.



Module 57-2: Config NASCOM DEBLOCKER MODULE Window

Module 57-3: NASCOM DEBLOCKER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Block Header Filter Bytes 1-12	Hexa-decimal	12 bytes	Defines the block filter criteria. The 12 entries in this field represent the first 12 bytes of the NASCOM Block header. To "pass" the input filter criteria, the header of an input block must exactly match any values entered in these fields. Blocks with headers that do not match the criteria are dropped (filtered) and are not processed by the module. Any field set to zero is not used for the filter criteria, i.e., those fields do not have to match. Default and standard filter criteria is 62 76 27 00 00 00 00 00 00 00 00 00.
Generate Echoes	Selectable Setup State	Enable Disable	Check box setup; insert check to enable. If enabled, every NASCOM block input to the module is output (echoed) to port 2. This does not change the deblocking output that occurs on port 1. Every block that is echoed has the following performed: source and destination values are swapped, it recalculates polynomial and puts it in the trailer, and modified block is output to port 2.
Reject CRC Errors	Selectable Setup State	Enable Disable	Check box setup; insert check to enable. When enabled, every input block with a CRC error is filtered. The polynomial is re-calculated for every input block and then compared to the value in the block trailer. If the two values do not match, this is considered a CRC error.
Hold Mode Enable	Selectable Setup State	Enable Disable	Check box setup; insert check to enable. If disabled, the block header hold bit (bit 66) is ignored. If enabled, then the hold mode bit is monitored and blocks are output according to the bit setting. If the bit is set (1), the deblock ed data is output immediately. If the bit is not set, the deblocker waits to receive a block with the bit set, and then outputs all deblocked data since the last hold bit set. Input blocks are held until the hold bit is set with the following exceptions: 3 second limit-- will not hold data for more than 3 seconds 5 block limit-- will not hold more than 5 blocks input before output
JSC Mode 1	Selectable Setup State	Enable Disable	Check box setup; insert check to enable. Mode developed to support JSC Mode 1 Commanding. The JSC Mode 1 Command Block header contains extra bytes (14 bytes instead of 12 bytes). When JSC Mode 1 is enabled, the Deblocker Module knows that the header is longer and strips it when deblocking
Delta Mode			Modified header for Delta Rifca.
Que Size	blocks		Defines how long the deblocker waits before beginning to output data. Deblocker will hold this number of blocks before it begins to transmit (applies to for the first time data is output only).

Parameter	Unit	Range or Values	Description
Frame Spacing	bytes		Created for SOHO. Forces a minimum inter-frame delay between uplinked commands. Does this by adding bytes to the end of the uplinked commands. The size set in this field is the number of bytes added to the end of each deblocked frame/command.

Extended Options Window Definition

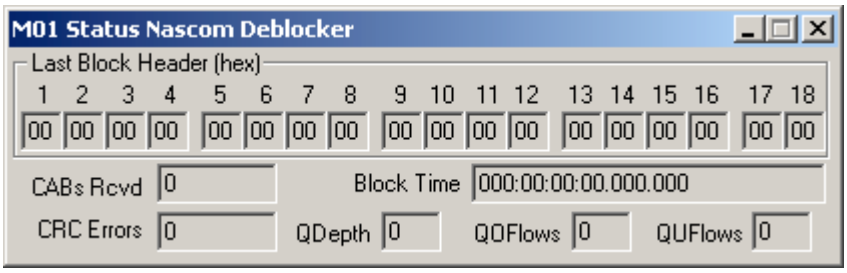
There is no extended options window for the NASCOM DEBLOCKER MODULE.

Configuration Notes

There are configuration notes for the NASCOM DEBLOCKER MODULE.

Status Window Definition

When you click the Status button on the NASCOM DEBLOCKER MODULE window, a Status NASCOM DEBLOCKER as shown below.



Module 57-4: Status NASCOM DEBLOCKER MODULE Window

Module 57-5: NASCOM DEBLOCKER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Last Block Header 1-18	Hexa-decimal	18 bytes	Displays first 18 bytes of the header of the last block received.
CABs Rcvd	count	Increments by one for each CAB input to module	Displays number of Circuit Assurance Blocks received. CAB is identified by a zero length field value; any block with zero length field is considered a CAB.
CRC Errors	count	Incrementing counter	Displays number of CRC Errors received. Increments by one for each block input that has a CRC error
Block Time		Reflects block header	Reports the time read from the last NASCOM Block Header input to the module.

Parameter	Unit	Range or Values	Description
Qdepth	blocks	Increments and decrements with queue depth	Number of Blocks currently in queue. Hold mode and Que size configuration affect this status. Displays current queue depth
QOFlows	blocks	N/A	Currently never triggers. However, a value in this field can be ignored.
QUFlows	blocks	N/A	Currently never triggers. However, a value in this field can be ignored.

Module 58 NASCOM RTP FORMATTER MODULE

MODULE NAME: NascomRTPFormatter.dll

Key Functions

- Socket module that outputs NASCOM Blocks with RTP header appended
- Provides a mapping table option, that reads a file to determine where to route a data unit based on its destination code field
- Routes according to destination code contained in input data unit: programmable destination code location in data unit
- Provides a mask for destination code field that makes only the defined bits pertinent to routing

Required Hardware

- No special hardware required for this module.

Add Module Guidelines

Do not exceed the Avtec PTP for Windows 32 modules per desktop limitation.

Required Input Source, Connections, and Events

INPUT SOURCE: Expecting input from the NASCOM Blocker, but source could be any module since the destination code index is programmable.

CONNECTIONS: Outputs to a socket. Outputs the same data sent to socket to port 1: data output with the header is also sent to port one.

EVENTS: Neither generates an event output nor requires an event input.

Module Functional Description

The NASCOM RTP FORMATTER MODULE accepts a NASCOM 4800 bit block and outputs an IP encapsulated frame. The interface for the NASCOM RTP FORMATTER MODULE is similar to the Network Sockets Module. The user inputs a Multicast IP address as the destination of the encapsulated frames. The Module then transmits all encapsulated 4800BB to that Multicast address.

This is a sockets module that outputs one of two protocols: UDP or multi-cast. The module accepts input data unit, prepends a 12-byte header, and then outputs to a socket.

A unique feature of this module is that it can be set up to use a mapping table. For this mode, the destination code that is read from the input data unit is output to a destination that is defined by the table (a file stored on the system).

Note: See **Appendix B** for a detailed overview of the NASCOM RTP header format.

Input Data Format

Accepts buffer input.

Output Data Format

Outputs the received buffer with the NASCOM RTP header appended.

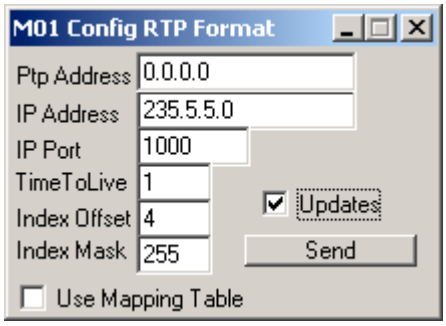
Module Window

When *NascomRTPFormatter.dll* is added to a desktop, the NASCOM RTP Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 58-1: NASCOM RTP FORMATTER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NASCOM RTP FORMATTER MODULE Board Parameters sub-menu.



Module 58-2: Config NASCOM RTP FORMATTER MODULE Window

Module 58-3: NASCOM RTP FORMATTER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
If Use Mapping Table is enabled, then IP Address and IP port are not used and only the values read from the mapping file are used.			
PTP Address	IP address	IP address	Address of local machine.

Parameter	Unit	Range or Values	Description
IP Address	IP address	IP address	Destination address of encapsulated 4800 BB. Class C or Class D address: for class C, send UDP protocol. For class D, send multicast IGMP (inter-group multi-class protocol) Class C is any address below 224.0.0.0 Class D is any address in the range of 224.0.0.0 through 239.255.255.255 inclusive
IP Port	Port number	1 to 65,535 some are reserved	IP Port of receiving machines. The destination port. Receiving machine port for UDP.
Time To Live	Header value	0 to 255	Number of hops before IP message dies Sets an IP header value. TTL defines how many routing devices this message goes through before expiring. Each router decrements an IP header value by one until this value is zero (expires)
Index Offset	byte	Data dependent	Defines Destination Address location in the header, which is the field to be used for table mapping Defines what byte to find the destination address at (this allows other types of data than NASCOM Blocks to be routed). Value entered is relative zero (first byte is zero): 4 is the standard location for a NASCOM Block.
Index Mask	decimal	0 to 255	Defines Mask for Index Offset This is the decimal equivalent of a hex mask. 255 means all bits are relevant. Setting a bit to zero means ignore that bit.
Use Mapping Table	Selectable Setup	Enable Disable	Enables the use of the MSS Mapping Table defined below When the mapping table is used, this means a Destination field in NASCOM Block Header is mapped to an IP address, and that is the address to which block is routed. To use this function, the software must find a text file in root directory of current drive: mss_table.dat. File has a specified format that was defined by NASCOM (see table format in configuration notes). The text file is parsed to find addresses.

Extended Options Window Definition

There is no extended options window for the NASCOM RTP FORMATTER MODULE.

Configuration Notes

MSS Destination Code Table (mapping table)

The Message Switching System (MSS) destination code table is stored in the file mss_table.dat in the root directory (i.e. c:\)

The MSS Table maps destination code of serial data block from/to MSS user to destination IP address for MSS CDs.

The following fields are contained in the MSS table:

- Destination Code Table Version Number
  1. *Definition:* Contains the version number of the Destination Code Table. The version number corresponds to a baselined data format which must be known by and compatible with the CD and NMS processes responsible for interpreting the file. Version number will be in the form of two decimal digits separated by a period (e.g.: 1.0).
  2. *Values:* Version numbers for the Destination Code Tables will be assigned and managed by NASCOM.
- MSS Destination Code
  1. *Definition:* Contains a valid MSS supported destination code.
  2. *Values:* 1 to 377 octal. Destination codes are assigned and managed by NASCOM
- Destination IP Address
  1. *Definition:* Contains the multicast address of either a single host or group of hosts to which the data is destined. The format is the standard dotted decimal notation for IP addresses comprised of four decimal integers separated by decimal points, where each integer gives the value o one octet of the IP address.
  2. *Values:* Destination IP addresses are assigned and managed by NASCOM
- Destination UDP Port
  1. *Definition:* Contains the UDP port number that the data is to be sent to on the receiving host specified by the Destination IP Address. Possible values for AUDP port numbers can range from 1025 to 65,535 decimal.
  2. *Values:* All Destination UDP Port numbers in the MSS Destination Code Table use 8001

The following table provides an example mapping table format:

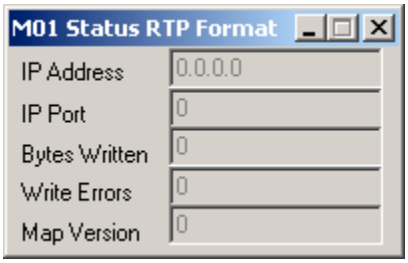
Module 58-4: MSS Destination Table Example

```
// MSS Front End Destination Code Table
//
// Version number
version 1.0
//
// Dest. Code      IP Address       UDP Port
  1               225.0.0.1        8001 // MILA Tracking
  2               225.0.0.2        8001 // MIL-71
  3               225.0.0.3        8001 // MIL-71 DSS-72
  4               225.0.0.4        8001 // Bermuda Tracking
                      :
                      :
  376             225.0.0.254      8001 // JPL-TOPEX
  377             225.0.0.255      8001 // WSC/ETGT
//
```

```
// end MSS Front End Dest Code Table
```

Status Window Definition

When you click the Status button on the NASCOM RTP FORMATTER MODULE window, a Status NASCOM RTP FORMATTER window appears, as shown below.



Module 58-5: Status NASCOM RTP FORMATTER MODULE Window

Module 58-6: NASCOM RTP FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
IP Address	IP address	IP address	The last transmitted destination-- reflects mapping table if enabled Displays receiver's IP Address
IP Port	Port	1 to 65,535 some are reserved	This field reflects the mapping table if enabled-- otherwise will match the values entered in config window Displays receiver's IP Port
Bytes Written	Bytes		Count of bytes written
Write Errors	Errors		Count of write errors
Map Version	Version		Reports mapping table version

Module 59 NASCOM RTP RECEIVER MODULE

MODULE NAME: NascomRTPReceiver.dll

Key Functions

- Accepts socket input from up to 16 different sources and outputs all data to same port
- Accepts Nascom Blocks with RTP Header prepended and then outputs Nascom Blocks with the RTP Header stripped

Required Hardware

- No special hardware is required.

Add Module Guidelines

A total of 32 modules may be added to one desktop. No other limitations apply to adding this module to a desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: Data input from one to 16 different socket connections.

CONNECTIONS: Outputs to port one.

EVENTS: Neither generates an event or expects an event input

Module Functional Description

The NASCOM RTP Receiver Module accepts IP encapsulated 4800BB from a network interface and outputs 4800BB. This Module can accept up to 16 different network streams and output each 4800BB stream to an independent output port.

This module is a 16-to-1 mux. It can receive input from up to 16 different ports and then output all data to the same port (port one).

The module is expecting Nascom Blocks with RTP header input from the socket connections. The RTP header is a 12-byte header**.** See Appendix B for a detailed overview of the NASCOM RTP format.

Input Data Format

Expects Nascom Blocks with 12-byte RTP Header prepended input.

Output Data Format

Outputs Nascom Blocks with RTP header stripped.

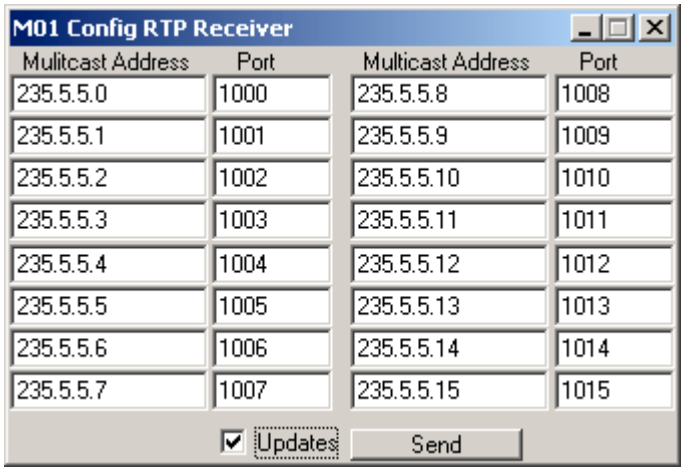
Module Window

When NascomRTPReceiver.dll is added to a desktop, the NASCOM RTP Receiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 59-1: NASCOM RTP RECEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NASCOM RTP RECEIVER MODULE Board Parameters sub-menu.



Module 59-2: Config NASCOM RTP RECEIVER MODULE Window

Module 59-3: NASCOM RTP RECEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Module is a 16 to 1 mux: it receives from all multicast and UDP addresses at the same time.			
Multicast Address	IP Address		Defines the Remote addresses from which data input is expected. Input the Multicast addresses to receive data streams If you don't want an address enter a zero and it comes back with 0.0.0.0
Port	Port		Input Port for each Multicast address Remote port

Extended Options Window Definition

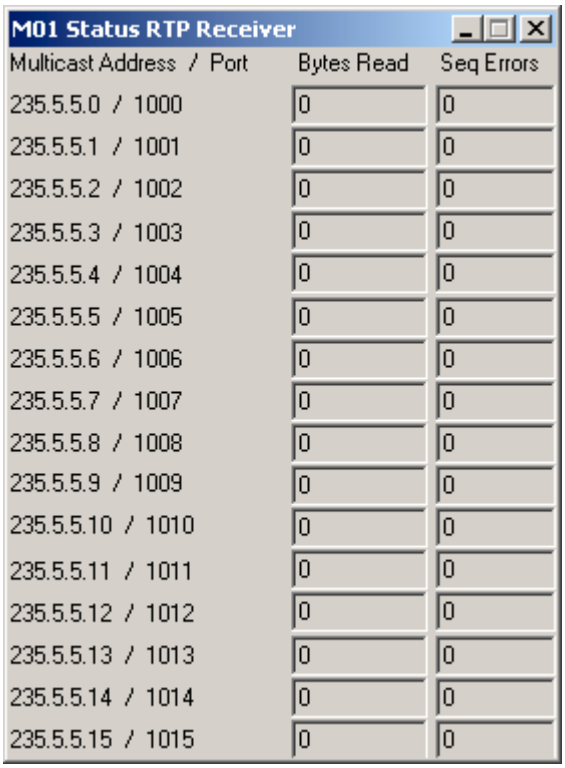
There is no extended options window for the NASCOM RTP RECEIVER MODULE.

Configuration Notes

There are no configuration notes for the NASCOM RTP RECEIVER MODULE.

Status Window Definition

When you click the Status button on the NASCOM RTP RECEIVER MODULE window, a Status NASCOM RTP RECEIVER window appears, as shown below.



Module 59-4: Status NASCOM RTP RECEIVER MODULE Window

Module 59-5: NASCOM RTP RECEIVER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Multicast Address/Port	IP address/ port n#		The values in these fields match whatever is set up in the configuration window.
Bytes Read	Bytes		Count of bytes read per input channel from that port and that address

Parameter	Unit	Range or Values	Description
Seq Errors	errors	Incrementing counter	Count of sequence errors per input channel The RTP header has a sequence counter that is a per virtual stream counter. This field reports a sequence error when the header does not increment by one. Sequence count of first block is valid and then from that point it should increment by one for each block received**.**

Module 60 NATIONAL INSTRUMENTS GPIB INTERFACE MODULE

MODULE NAME: GPIB.dll

Key Functions

- Generic Module to control a GPIB remote control device

Required Hardware

- AT-GPIB/TNT ISA card.

Add Module Guidelines

One module can control any number of GPIB devices.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives user input GPIB directives from the console gui or via the Remote Interface Library (RIL) calls. The directive input is always ASCII text. The user may also define script files containing predefined directives to send. See the GPIB functional description for the exact specification.

CONNECTIONS: No module connections should be made with this module. A GPIB cable is required to establish connectivity between the DCS unit and the GPIB card installed in the computer. See the National Instruments documentation (enumerated below) for cable length restrictions, and correct procedures for establishing a GPIB network.

EVENTS: No events are expected or generated.

Module Functional Description

The GPIB Module supports the National Instruments General Purpose Interface Bus (GPIB) Board. GPIB is a standard method used to control instruments and takes measurements remotely. The GPIB module provides a generic interface to test the remote control capability of other instruments. This module can take user input and send it to a specific device, and await and accept a response or data from the instrument. This module is not tailored toward any specific instrument; rather it can address all instruments. Note: The Module Status for the GPIB Module is displayed in the Configuration window.

The actual electrical interface is a National Instruments GPIB (General Purpose Interface Bus) NI-488.2. The card used is model AT-GPIB/TNT. It is an ISA-bus based card. An interrupt, I/O port address, and DMA channel all need to be set using jumpers on the card.

The default card settings are: IRQ 11, I/O 2C0, and DMA 5. The DMA channel can be disabled if desired; it is not used by this module. Refer to the National Instruments documentation supplied with your GPIB card. This module was coded using the October 1996 edition of "Getting Started with Your AT-GPIB/TNT and the NI-488.2M Software for Windows-NT", part number 320648C-01. The GPIB function calls for this module were defined using the December 1996 edition of "NI-488.2M Function Reference Manual for Win32", part number 321038C-01.

Input Data Format

Input must be valid GPIB commands. The National Instruments documentation provides general syntax definitions; the exact options and syntax to be used varies and is defined by the specific piece of hardware.

Output Data Format

ASCII output responses can be received and displayed from the remote hardware. The actual format is defined by the hardware.

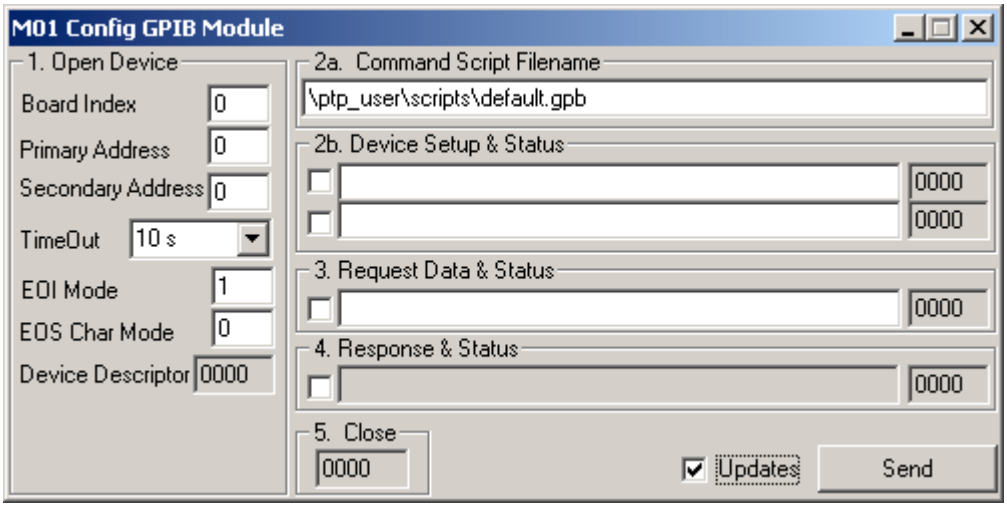
Module Window

When GPIB.dll is added to a desktop, the GPIB dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 60-1: GPIB MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure GPIB MODULE Board Parameters sub-menu.



Module 60-2: Config GPIB MODULE Window

Module 60-3: GPIB MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
1. OPEN DEVICE			
Board Index	Decimal	Not restricted; 0 denotes the first GPIB board in the system.	Board number of GPIB. One GPIB board controls a GPIB network of devices.
Primary Address	Decimal	0 to 31	Address of GPIB Device that is to be configured via the GPIB board selected by the Board Index parameter
Secondary Address	Decimal	0 to 31	Secondary Address of GPIB Device (not used by all GPIB Devices)
Time Out	User Selectable		Read/Write timeout for Device; User-selectable from 10 us to 1000 s
EOI Mode			End of Input Mode – defines how to terminate commands sent from GPIB Board to GPIB Device; Default terminator = 0
EOS Char Mode			End of String Character Mode – defines how to terminate strings sent from GPIB Board to GPIB Device; Default terminator = 0
Device Descriptor	Decimal	Display Only	Device Descriptor returned from the GPIB open call. This value is used to reference the device until it is closed.
2A. COMMAND SCRIPT FILENAME			
filename	ASCII	Valid path and filename	Filename of text file that contains commands to be sent to GPIB device. The text file can be up to 30 lines long; each line is of the form: *[name of command]=[string]*. These commands will be listed in the Command GPIB Module drop-down list.
2B. DEVICE SETUP & STATUS			

Parameter	Unit	Range or Values	Description
	ASCII	Valid GPIB command	Commands to be sent to GPIB device. To send command, check checkbox and click Send in the Configuration window. Grayed edit box to right of command displays status. Two lines are available, either or both can be used. GPIB status is returned in the gray window at the right.
3. REQUEST DATA & STATUS			
	ASCII	Valid GPIB command which requests information from the remote device	The command in the first line is sent to the remote device. If the device returns data, it is displayed in area 4. Check the checkbox to enable this command. GPIB status is returned in the gray window to the right.
4. RESPONSE & STATUS			
	ASCII	Display Only	Displays responses to any commands and requests sent in area 3.
5. CLOSE			
	Decimal		Close status for GPIB device.

Extended Options Window Definition

There is no extended options window for the GPIB MODULE.

Configuration Notes

When you click the Command button on the GPIB MODULE window, a Command GPIB window appears, as shown below.

Module 60-4: Command GPIB MODULE Window

Module 60-5: GPIB MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected		Name of each command	Displays available commands to send to GPIB device. Commands are based on commands listed in file referenced by *Commands Script Filename* in the Config GPIB Module window. If "No Command Selected" is selected, input is effectively no-oped.

Status Window Definition

There is no status window for the GPIB MODULE.

Module 61 NETWORK SOCKETS MODULE

MODULE NAME: NetworkSockets.dll

Key Functions

- Establishes and transmits or receives data on a network connection
- Supports high latency networks
- Provides UDP, TCP client, TCP server, and Multicast connections
- Supports multi-homing

Required Hardware

- NETWORK SOCKETS MODULE does not require any special hardware be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of NETWORK SOCKETS MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: For transfer to network, any module can feed the socket data. A Spooler Module may be necessary for conditions that require rate buffering. For receiving data from the network, data source is the socket connection.

CONNECTIONS: When outputting data to network, data is lost unless the connection is established and maintained.

EVENTS: Generates events that can be fed back to a data source. Events occur at the maximum data rate that the network supports. Events are regenerative, which means that as a default it always generates an event once a second.

Module Functional Description

The NETWORK SOCKETS MODULE supports data transfer to/from a network interface. The Sockets Module supports connection oriented data transfer using TCP/IP sockets. The Sockets Module supports connectionless data transfer using UDP/IP sockets. The Sockets Module also support IP multicast so that data can be transferred to multiple hosts simultaneously without multiple transmissions on the network. Note: IP multicast may not be supported by all routers.

Input Data Format

Any data format can be input.

Output Data Format

This module outputs whatever data was input.

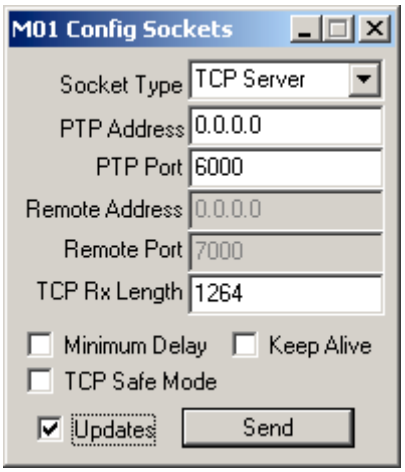
Module Window

When *NetworkSockets.dll* is added to a desktop, the Network Sockets dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 61-1: NETWORK SOCKETS MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NETWORK SOCKETS MODULE Board Parameters sub-menu.



Module 61-2: Config NETWORK SOCKETS MODULE Window

Module 61-3: NETWORK SOCKETS MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Socket Type	Selectable Setup	UDP TCP Server TCP Client Multicast	Choice of UDP, TCP Server, TCP Client, or Multicast If set to server, it opens up the port and waits for connection. If set to client, connects to a server that has already opened port. In client mode, it keeps trying to connect.

Parameter	Unit	Range or Values	Description
PTP Address	Address	IP address	Local Avtec PTP for Windows address. Address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	Port number	1 to 65,535 some are reserved	The local socket that is to be opened. Socket number (use for TCP Server or UDP Receiver)
Remote Address	Address	IP address	Remote IP address (only used for TCP client, Multicast or UDP transmit) Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal.
Remote Port	Port number	1 to 65,535 some are reserved	Remote Socket number (only used for TCP client, Multicast, or UDP transmit) Defines the socket on the remote machine on which it expects data transfer
TCP Rx Length	bytes		TCP Receive Length in bytes To see frame aligned data, this must be a multiple of the expected frame length input. If it is not a multiple, then data will be lost on the last frame.
Minimum Delay	Selectable Setup	Enable Disable	Applies only to TCP modes. When enabled, the protocol stack buffer size is 512 kB. It also disables the Nagle algorithm. Designed for high latency networks.
TCP Safe Mode	Selectable Setup	Enable Disable	Applies only to TCP modes. If enabled, emphasis the transmission integrity. Blocks the data source module from sending more data if the socket module is not ready. Prevents the network pipe from being overflowed.
Keep Alive	Selectable Setup	Enable Disable	Applies to TCP only. Network integrity test. Uses an inactivity timer that starts when no data has been sent/received on the connection. Used to decide if the connection is broken. Works like a ping. It sends an Out of Band message on the socket and expects a return message also on the Out of Band. If Keep Alive message is not answered: client tries to reconnect and server disconnects everyone and then waits for a reconnect.

Extended Options Window Definition

There is no extended options window for the NETWORK SOCKETS MODULE.

Configuration Notes

User Datagram Protocol (UDP) is a connectionless transport protocol. It allows applications to send and receive encapsulated raw IP (Internet Protocol) data without having to establish a connection. This protocol is widely used when quick delivery of data is more important than accurate delivery, as in real-time applications.

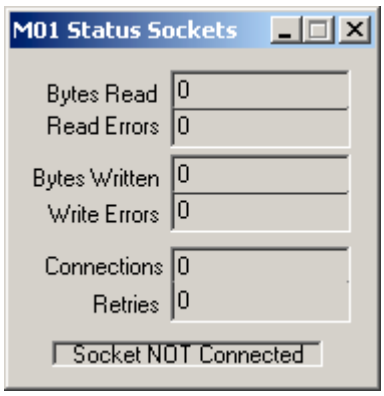
IP Multicasting can be used when you want a specific data stream to be directed to multiple users on a specific network group. Multicasting is a subset of the UDP protocol. This means that it is a connectionless transport protocol and does not attempt to establish a connection. It simply sends data to a Multicast IP address. Multicast addresses are of the class D IP address format. Class D addresses fall in the range 224.0.0.0 through 239.255.255.255. Each class D address identifies a group of hosts and 28 bits are assigned for identifying groups. This means that over 250 million groups can exist at any one time. When a packet is sent to a multicast address, a best-efforts attempt is made to deliver that packet to all the members of the group that was addressed. However, there are no guarantees on delivery, thus some members of the group may not receive the packet.

Transmission Control Protocol (TCP) is a connection-oriented transport protocol. This means that a handshake, or an acknowledgment, takes place between the transmitter (server) and the receiver (client). It is designed to provide a reliable byte stream over unreliable networks and internetworks. It fragments an incoming data stream into discrete messages and sends them to a specified location. At the destination location, the receiving TCP process reassembles the received messages into its original sequence. The TCP process also handles flow control to prevent a fast sender from overloading a slow receiver with more messages than it can handle.

Configuring the NETWORK SOCKETS MODULE appropriately depends on whether the module will be used as a transmitter or a receiver and the *Socket Type* selected in the configuration window. After these requirements are determined, we can configure the Avtec PTP for Windows *Address, Port, Remote Address, Remote Port,* and *TCP Rx Length*. The *TCP Rx Length* is usually set to the frame length.

Status Window Definition

When you click the Status button on the NETWORK SOCKETS MODULE window, a NETWORK SOCKETS window appears, as shown below.



Module 61-4: Status NETWORK SOCKETS MODULE Window

Module 61-5: NETWORK SOCKETS MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes		Total bytes received on the socket. Count of bytes read
Read Errors	Errors		Count of read errors Read error = socket failure--a request to read bytes from the host is meet with a network error. Module re-initializes when a read error is received.
Bytes Written	Bytes		Total number of bytes transmitted to the socket. Count of bytes written
Write Errors	Errors		Count of write errors Write error = tried to write to a host and received a network error. If trying to transmit more than network can accommodate, will receive a write error. Only occurs, if safe mode is off or network goes down.
Connections	Connections		Current number of connections. Number of times a successful connection has been made includes re-connections.
Retries	Retries		Number of connection retries-- number of times a connection was attempted

Module 62 NETWORK SOCKETS VERSION 2 MODULE

MODULE NAME: NetworkSocketsVer2.dll

Key Functions

- Provides bi-directional TCP server socket connection
- Receives data on the same socket from multiple clients
- Allows multiple client connections on same socket to provide same data to multiple destinations

Required Hardware

- This module does not require any special hardware installed in Avtec PTP for Windows system.

Add Module Guidelines

Do not exceed the limitation of 32 modules per desktop.

Required Input Source, Connections, and Events

INPUT SOURCE: The input data source can be any other module or a socket. It is a bi-directional socket.

CONNECTIONS: Output is to the socket or to port one depending on data direction.

EVENTS: This module neither generates an event nor requires an event input.

Module Functional Description

The NETWORK SOCKETS VERSION 2 MODULE, similar to the Network Sockets Module, supports data transfer to/from a network interface. As opposed to the Sockets Module, the NETWORK SOCKETS VERSION 2 MODULE only supports these transfers in a TCP Server mode. However, the TCP Server mode for the Sockets Version 2 supports multiple clients (the Sockets Module supports one TCP client). This module allows multiple clients to request an identical data stream from a single Server with the reliability of a TCP connection.

The NETWORK SOCKETS VERSION 2 MODULE outputs to the socket at whatever rate data is input. Each socket connection has a 6 frames/buffer queue. If client overruns the queue and protocol stack, then data is lost if client cannot keep up.

Input Data Format

Any data format can be input.

Output Data Format

Data is throughput: whatever is input is throughput.

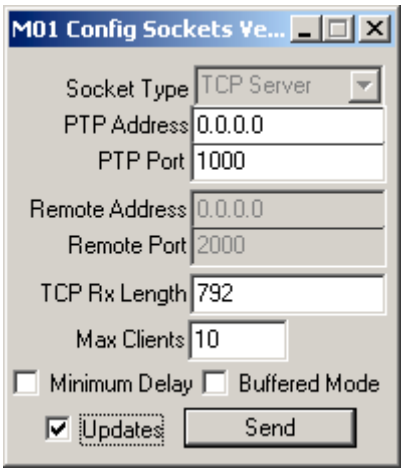
Module Window

When *NetworkSocketsVer2.dll* is added to a desktop, the Network Sockets Version 2 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 62-1: NETWORK SOCKETS VERSION 2 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NETWORK SOCKETS VERSION 2 MODULE Board Parameters sub-menu.



Module 62-2: Config NETWORK SOCKETS VERSION 2 MODULE Window

Module 62-3: NETWORK SOCKETS VERSION 2 MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Socket Type	N/A	N/A	Cannot be configured. It is always in TCP Server mode.
PTP Address	IP Address		Cannot be configured. Defaults to IP address of local machine. Not user-changeable.

Parameter	Unit	Range or Values	Description
PTP Port	Port number		User selectable socket port to be used for TCP transfers. This is the port on the machine on which the module is running
Remote Address	N/A	N/A	Not used
Remote Port	N/A	N/A	Not used
TCP Rx Length	Bytes		TCP Receive Length in bytes Defines the data portion of the message and the buffer size output on port one.
Max Clients	Clients	1 to 32	Sets maximum TCP Clients to allow
Minimum Delay			When enabled, disables TCP nagle algorithm and sets socket priority to highest. Insert a check to enable.
Buffered Mode			Data is buffered for each client, where data coming in is faster than the data being read out.

Extended Options Window Definition

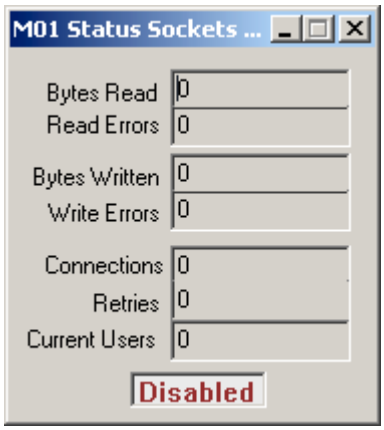
There is no extended options window for the NETWORK SOCKETS VERSION 2 MODULE.

Configuration Notes

There are no configuration notes for the NETWORK SOCKETS VERSION 2 MODULE.

Status Window Definition

When you click the Status button on the NETWORK SOCKETS VERSION 2 MODULE window, a Status NETWORK SOCKETS VERSION 2 window appears, as shown below.



Module 62-4: Status NETWORK SOCKETS VERSION 2 MODULE Window

Module 62-5: NETWORK SOCKETS VERSION 2 MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	N/A	Not Used
Read Errors	Errors	N/A	Not Used
Bytes Written	Bytes	Incrementing Count	Count of bytes written
Write Errors	Errors	Incrementing Count	Count of write errors
Connections	Connections	Count	Number of connections
Retries	Retries	Incrementing Count	Number of connection retries
Current Users	Users	Count	Number of clients currently connected to server

Module 63 NULL TRANSCEIVER MODULE

MODULE NAME: NullTransceiver.dll

Key Functions

- Snapshots and displays the current input buffer
- Performs no data processing
- Provides configurable display size and configurable offset into display
- Displays in hex, decimal, text, binary, or octal

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of NULL TRANSCEIVER MODULES can be added to a desktop as long as the 32-module limit per desktop is not exceeded.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module can provide the input data.

CONNECTIONS: Data is output on port 1. Unless the output port is connected to another module's input port, data is dropped.

EVENTS: This module neither generates an even output or requires an event input.

Module Functional Description

The NULL TRANSCEIVER MODULE is used to snapshot data: it performs no data processing. It views input data buffers and passes the data through unaltered. Configuration provides the operator an offset into the data buffer so that the view can start at a specific point in the data. A view length in bytes is also provided.

The module is primarily used for debugging, but it can also be used to multiplex several data streams into a single data stream.

Input Data Format

Any buffer data format can be input.

Output Data Format

Module displays exactly the data input. However, the display shows only the portion of the input buffer that is defined by configuration. Check offset and display size values if display does not reflect the expected data (especially if the display is short by a number of bytes).

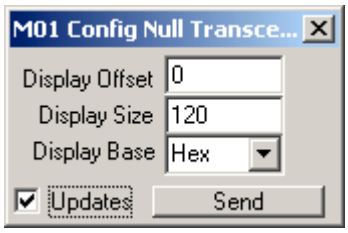
Module Window

When NullTransceiver.dll is added to a desktop, the Null Transceiver dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 63-1: NULL TRANSCEIVER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure NULL TRANSCEIVER MODULE Board Parameters sub-menu.



Module 63-2: Config NULL TRANSCEIVER MODULE Window

Module 63-3: NULL TRANSCEIVER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Display Offset	Bytes	0 to buffer size	Defines the offset into the buffer to be displayed. The number of bytes entered here is truncated from the beginning of the data buffer for display purposes only. The data remains unaltered, but the display does not show the offset bytes.
Display Size	Bytes		Defines the total number of bytes that is displayed in the window. If this size is less than the buffer size, then only a part of each buffer is displayed in the window.
Display Base	Setup	Hex Decimal Octal Text Binary	Defines the unit in which the window displays

Extended Options Window Definition

There is no extended options window for the NULL TRANSCEIVER MODULE.

Configuration Notes

When not sure how big (size) to make the NULL TRANSCEIVER MODULE window, set the size bigger than needed. Too small of a size may cause results to be misinterpreted. Since the window truncates the input data buffers at the setup size, more data may be in a buffer, but not displayed because the setup size is too small.

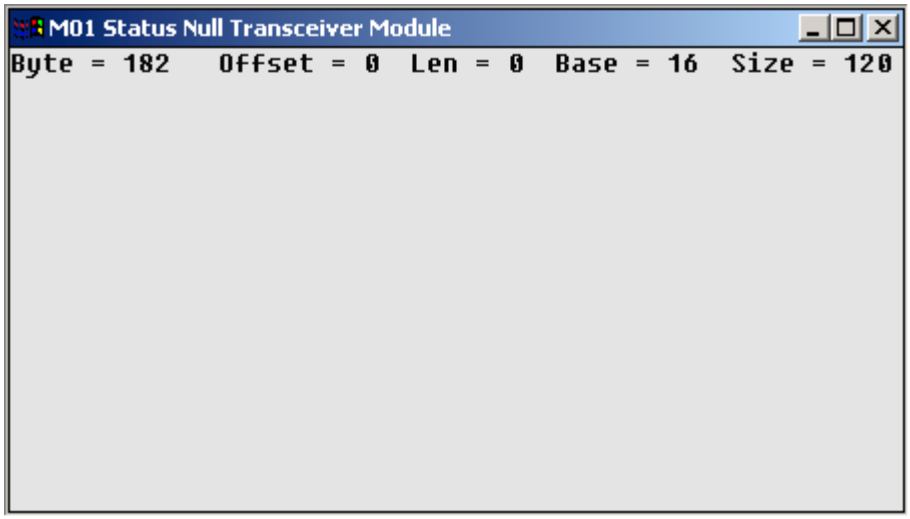
Remember that the data shown in the status window is not necessarily all the data there is to see. To interpret the status display, open the configuration window to view how the module is configured. The value of the *Display Offset* determines which byte of the frame the Status window will show first. The *Display Size* determines how many bytes after the display offset will be shown in the display window. For example, if *Display Offset* is set at 140 and *Display Size* is set at 120 with a *Display Base* of Hex, the Status window shows bytes 140 through 260 in hexadecimal. In this example, to view the Reed-Solomon bytes appended at the end of the frame, (depending on frame size) it may be necessary to adjust the offset and display size.

Status Window Definition

When you click the Status button on the NULL TRANSCEIVER MODULE window, a Status NULL TRANSCEIVER window appears, as shown below.

The status window displays the raw data from the buffers that are sent to the NULL TRANSCEIVER MODULE input port. The data is extracted from the buffer at the location defined in the configuration parameters (offset field).

The configuration parameters (offset, len, base, and size) are displayed along the top of the Status window. In addition, a *Byte* = field is available. This field displays the byte position of the value that the mouse pointer is over. The Status window can also be resized.



Module 63-4: Status NULL TRANSCEIVER MODULE Window

Module 64 ODETICS GPS TIME BOARD MODULE

MODULE NAME: OdeticsTime.dll

Key Functions

- Control and Status reporting for the Odetics GPSync-ISA Time card.
- Capability to set system time from the Odetics board.

Required Hardware

- The following card must be installed in the Avtec PTP for Windows system:
  - ✓ Odetics GPSync-ISA Time Board

Add Module Guidelines

Only one Odetics board per system. One module to control it.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives data from an external SNA connector. The card is cabled using RG-58 cable to a small egg-shaped, standalone GPS antenna.

CONNECTIONS: No module connection should be made to this module.

EVENTS: No events are generated or expected.

Module Functional Description

The installation of the Odetics driver (provided by Odetics) and card create an additional communications port, typically COM3. The user should check the BIOS setting in the computer being used, and enable COM3 (or whatever is the next unused COM port) if necessary.

This module reads from the new communications port to obtain the current time. The user may use the Control window to set the PC time. At load time, the module will automatically attempt to set the system. This automatic attempt could fail, or could set the time to an incorrect value if the antenna is not connected or the card has the incorrect time. When no antenna is connected the card continues to provide time from an internal oscillator. The card re-synchronizes the time when it achieves solid lock on three GPS satellites.

A cold start (no ephemeris data available) of the unit takes about six hours according to the Odetics documentation. Warm starts require at least twenty minutes to report valid time and location data.

The interface through the COM port is rather cumbersome at best. To aid the user, this module automatically sends commands to the board. These commands request the value of

the most used data points available from the card. The response is parsed and translated. The responses are available from the Status gui or from the Avtec PTP for Windows using Remote Interface Library (RIL) function calls. The user should become familiar with the standard format of commands and responses from the card, using the manuals supplied with the card. The user may insert his own Odetics commands and view the responses in the Status window or via RIL function calls. These can be used to obtain information from the board that is not displayed on the gui. An example of a command requiring user input / output is the Odetics capability to latch events.

Input Data Format

Any user input is in text format. The user syntax is published in the supplied Odetics manual.

Output Data Format

The output is directed to the Config window gui and is also available using RIL function calls.

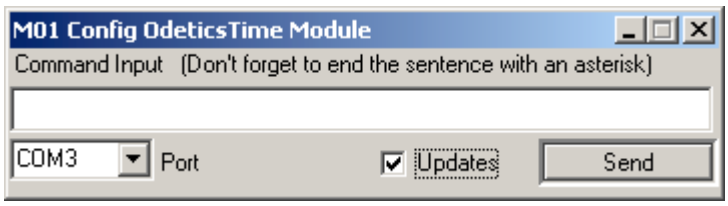
Module Window

When OdeticsTime.dll is added to a desktop, the Odetics GPS Time Board dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 64-1: ODETICS GPS TIME BOARD MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure ODETICS GPS TIME BOARD MODULE Board Parameters sub-menu.



Module 64-2: Config ODETICS GPS TIME BOARD MODULE Window

Module 64-3: ODETICS GPS TIME BOARD MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Command Input	Text	Any Syntactically correct Odetics command	User supplied command. The user command is interspersed with the commands generated by this module. All responses are reported in the status window.
Port	Selectable Setup	COM1, COM2, COM3, COM4	Select the communications port that the Odetics hardware and driver have created.

Extended Options Window Definition

There is no extended options window for the ODETICS GPS TIME BOARD MODULE.

Configuration Notes

When you click the Command button on the ODETICS GPS TIME BOARD MODULE window, a Command ODETICS GPS TIME BOARD window appears, as shown below.

Module 64-4: Command ODETICS GPS TIME BOARD MODULE Window

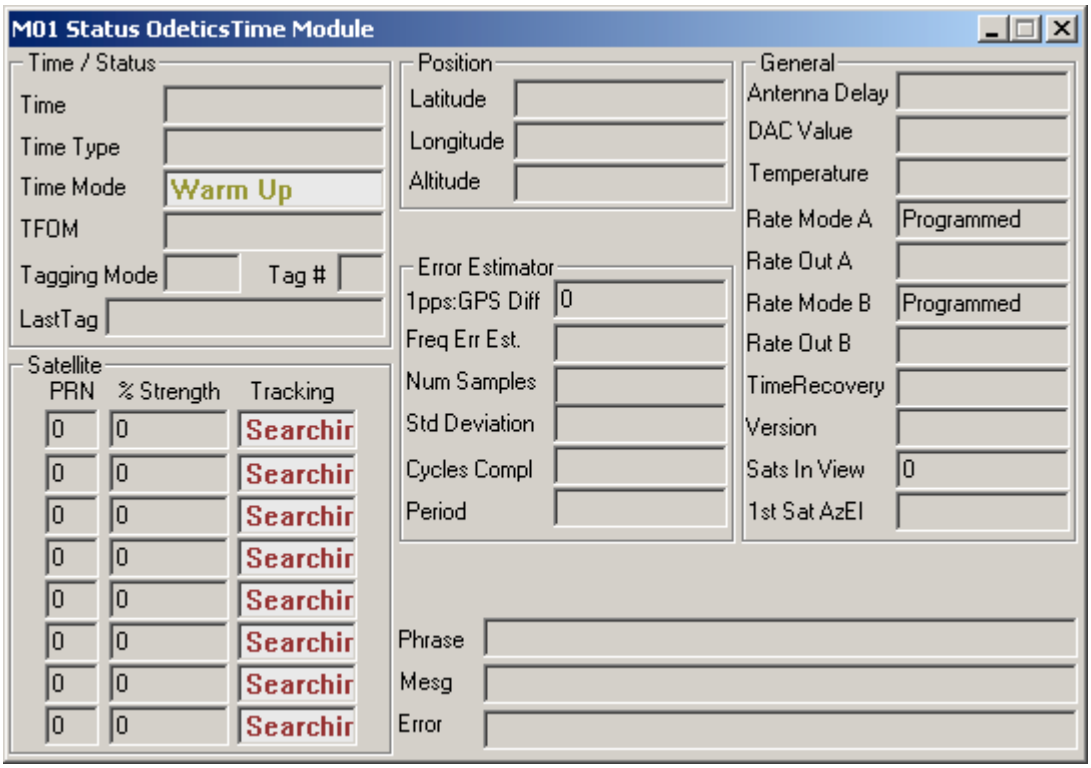
Module 64-5: ODETICS GPS TIME BOARD MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	N/A		No command can be output when this listbox entry is selected.
Recent Error	N/A		Sends a command to the board requesting recent error data.
Warm Start	N/A		Sends a command to the board instructing it to perform an immediate warm start.
Cold Start	N/A		Sends a command to the board instructing it to perform an immediate cold start.
Clear All Tags	N/A		Sends a command to the board instructing it to immediately clear all accumulated time tags. Tag accumulation is the responsibility of the user.
Tag Buffer Wrap	N/A		Sends a command to the board instructing it to wrap the tag buffer when necessary. The buffer contains 100 tags.
Tag Buffer Burst	N/A		Sends a command to the board instructing it to stop collecting tags when full.

Parameter	Unit	Range or Values	Description
Time Recov Dynamic	N/A		Sends a command to the board instructing it to set the Dynamic Recovery mode.
Time Recov Position Avg	N/A		Sends a command to the board instructing it to set the Position Average mode.
Time Recov Unknown	N/A		Sends a command to the board instructing it to set the Recovery Unknown mode.
Set System Time	N/A		Sets the system time from the time read through the communications port. The accuracy of this time is to the nearest second.

Status Window Definition

When you click the Status button on the ODETICS GPS TIME BOARD MODULE window, a Status ODETICS GPS TIME BOARD window appears, as shown below.



Module 64-6: Status ODETICS GPS TIME BOARD MODULE Window

Module 64-7: ODETICS GPS TIME BOARD MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
TIME/STATUS			
Time			Time received from the board.
Time Type			Type of time reported. Examples are GMT, Local.

Parameter	Unit	Range or Values	Description
Time Mode			Displays Warm Up or Time Lock status.
TFOM			
Tagging Mode			
Tag#			
Last Tag			Last Time tag stored on the board.
SATELLITE			
PRN			Satellite number. Ranges from 1 to 31.
% Strength			Signal strength from a satellite. This value is reported in the nearest 10%.
Tracking			Displays Search or Lock status
POSITION			
Latitude			Antenna location latitude.
Longitude			Antenna location longitude.
Altitude			Antenna location altitude.
ERROR ESTIMATOR			
1pps:GPS Diff			
Freq Err Est.			
Num Samples			
Std Deviation			
Cycles Comp			
Period			
GENERAL			
Antenna Delay			
DAC Value			
Temperature			
Rate Mode A			
Rate Out A			
Rate Mode B			
Rate Out B			
Time Recovery			
Version			
Stats In View			
1st Sat AzEl			
OTHER SETUP			
Phase			Text input to the Odetics board to control setup or request status.
Mesg			Response from the Odetics board.

Parameter	Unit	Range or Values	Description
Error			Last error message from the Odetics board.

Module 65 PACKET PROCESSOR MODULE

MODULE NAME: PacketProcessor.dll

Key Functions

- Reassembles packets from input CCSDS version 1 and version 2 frames
- Optionally strips packet headers
- Optionally outputs all reassembled packets (no filter criteria)
- Filters packets for output based on APID
- Lists all identified APIDs found during a data flow

Required Hardware

- This module does not require that any special hardware be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of PACKET PROCESSOR MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops. Data rate and frame length do affect PACKET PROCESSOR MODULE processing

Required Input Source, Connections, and Events

INPUT SOURCE: Data input should be from either the Virtual Channel Processor or the Virtual Channel Processor Version 2 modules. Data can be input from the Serial Input Module, but the input would not include Reed-Solomon or virtual channel statistics.

CONNECTIONS: Outputs to port one.

EVENTS: Neither generates an event or requires an event input. It will ignore any event input.

Module Functional Description

The PACKET PROCESSOR MODULE extracts packets from a single virtual channel of a CCSDS telemetry stream. It receives a stream of transfer frames and outputs a stream of packets. The user defines a list of Application IDs (APID) and the Packet Processor will strip packets with matching APIDs to its data output for another module. In general, the Packet Processor is preceded by the CSDS Virtual Channel Processor Module which sorts frames based on virtual Channel ID.

Input Data Format

Input is either CCSDS Version 1 or Version 2 frames. Ideally, input frames were processed by a VC processor prior to input so that the frame includes RS and virtual channel statistics.

Output Data Format

Output is version 1 packets. Each packet may or may not include the packet header depending on module configuration.

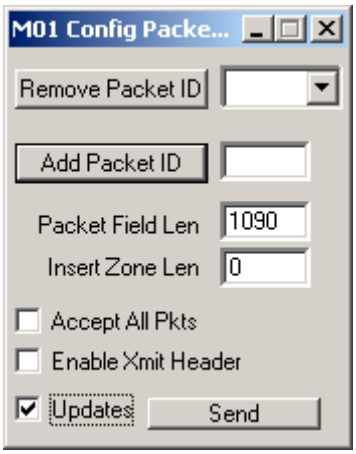
Module Window

When *PacketProcessor.dll* is added to a desktop, the Packet Processor dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics. Left mouse click Command to view the command window, which is currently not implemented.

Module 65-1: PACKET PROCESSOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure PACKET PROCESSOR MODULE Board Parameters sub-menu.



Module 65-2: Config PACKET PROCESSOR MODULE Window

Module 65-3: PACKET PROCESSOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Remove Packet ID	Packet APID values in decimal	Any packet entered with "Add Packet ID"	The list available through the down arrow provides APIDs of all packets that will be recognized and output from this module. The "Remove Packet ID" command allows the operator to remove a packet APID from the output packet list. To remove an APID from then list of output packets, select the APID from the list, wait for the selected APID to appear in the window, and then issue the "Remove Packet ID" command. The list of APIDs available in the window are those that have been entered with "Add Packet ID."
Add Packet ID	Packet APID values in decimal	APID field is 11-bits	This field allows the operator to define which packets, by APID value, are reassembled and output from the module. For packet filtering, use this entry field to enter only those packets APID values that are to be reassembled. For each APID, enter the APID value in the window and then issue the Add Packet ID command. Once a packet is entered in the output list, the APID value shows in the list to the right of the Remove Packet ID command. Maximum of 50 APIDs can be added.
Packet Field Len	bytes	0 to Frame length - (sync, header, CRC, CLCW, insert zone, and RS symbols)	Defines the number of bytes in the input frame that contains packet data: defines the length of the Packet Data Field within the Transfer Frame. The field begins after the first header pointer and continues to either the CRC (if included), CLCW (if included) or Reed-Solomon symbols. Note for reference purposes: version 1 frames have a 10 byte header including sync; version 2 frames have a 12 byte header including sync. Account for an insert zone if present.
Insert Zone Len	bytes	0 to Frame length minus (sync, header, CRC, CLCW, and RS symbols)	Needs to be set up to match the input frame format. If the input frame includes an insert zone, the insert zone length must be defined here. Set to zero for no insert zone. The insert zone precedes the first header pointer.
Accept All Pkts	Selectable Setup	Enable Disable	When enabled, all input packets are reassembled and output from the module regardless of the packet APID. Checkbox setup: insert a check to enable and remove check to disable.
Enable Xmit Header	Selectable Setup	Enable Disable	Defines how reassembled packets are output from the module. When enabled, each packet is output with the packet header still attached. When disabled, the packet header is stripped from each packet prior to output from the module. Checkbox setup: insert a check to enable and remove check to disable. Pkt header is 6 bytes for version 1 packets Pkt header is 14 bytes for version 2, but even if disabled, only 6 bytes are stripped prior to output

Extended Options Window Definition

There is no extended options window for the PACKET PROCESSOR MODULE.

Configuration Notes

Although the PACKET PROCESSOR MODULE includes a command window, the commands are not implemented and the window currently provides no functionality.

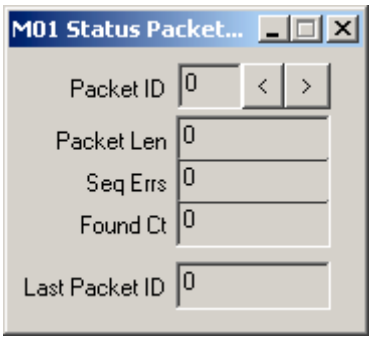
Module 65-4: Command PACKET PROCESSOR MODULE Window

Module 65-5: PACKET PROCESSOR MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description
No Command Selected	N/A	N/A	Disables command function

Status Window Definition

When you click the Status button on the PACKET PROCESSOR MODULE window, a Status PACKET PROCESSOR window appears, as shown below.



Module 65-6: Status PACKET PROCESSOR MODULE Window

Module 65-7: PACKET PROCESSOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
Packet ID	APID	All APIDs detected in the input stream	This field allows the operator to chose which APID statistics will be displayed in the rest of the status window. To choose the APID, use the right/left arrows provided. The available selections are the APIDs defined by the config window for re-assembly. The selection made from this list of packet APIDs defines the APID for which all other APID-specific status in the window applies. To change the display and view status for a different APID, select an APID from the available list-- that APID shows in the window once it is selected. This list reflects the APIDs that have been detected in the input data stream: if an APID is not available in the list, then that means it hasn't detected any packets of that APID.
Packet Len	bytes	variable	Displays the detected packet length for that APID. Displays the exact information read from the packet header length field. Since the value is read directly from the packet header, the display length is packet length minus one. Displays the Packet length for last packet received with the APID chosen in "Packet ID."
Seq Errs	count	Increments by one for each sequence error	Displays the number of sequence errors that have occurred for the APID chosen in "Packet ID." Reflects the number of discontinuities.
Found Ct	count	Increments by one for each packet input	Number of packets reassembled and output with the APID chosen in "Packet ID."
Last Packet ID	APID	11-bit field	Reflects the APID of the last packet input to the module. Accuracy of status depends on display update rate.

Module 66 PACKET PROCESSOR VERSION 2 MODULE

MODULE NAME: PacketProcessorVer2.dll

Key Functions

- Reassembles packets from input CCSDS version 1 and version 2 frames
- Optionally deletes fill packets
- Optionally strips packet headers
- Optionally outputs all reassembled packets (no filter criteria)
- Filters packets for output based on APID, packet length, and packet version
- Provides more extensive filter criteria and status than Packet Processor Module. Filter criteria defined on a per APID basis.
- Lists all identified APIDs found during a data flow (see config window, maximum 64 APIDs)
- Optionally deletes partial packets

Required Hardware

- This module does not require any special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

Any number of PACKET PROCESSOR VERSION 2 MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Expects frame input from Virtual Channel Processor Ver 2 Module.

CONNECTIONS: Outputs to port 1. Another module's input port must be selected to accept the output or the data is dropped although statistics are still maintained.

EVENTS: This module neither generates any events or requires an event input.

Module Functional Description

The PACKET PROCESSOR VERSION 2 MODULE reassembles packets from CCSDS transfer frames and CADUs (CCSDS version 1 and version 2 frames). This module receives a stream of frames and outputs a stream of packets. The module can be configured to extract all packets from the input stream or it can be configured to output only a selected list of packets based on packet Application Process Identifier (APID), packet version number, and packet length.

The PACKET PROCESSOR VERSION 2 MODULE adds control and status features that are not available in the Packet Processor Module. The additions were developed specifically to

support the DataLynx project. The additional features include the option to delete fill packets and the ability to define packet re-assembly by APID, packet version number, and packet length (as opposed to definition by APID only).

The PACKET PROCESSOR VERSION 2 MODULE can process packet headers that have –1 for first header pointer, adds multiple filtering options, and allows you to delete partial packets.

Input Data Format

Input is either CCSDS Version 1 or Version 2 frames. Frames should have been processed by the Virtual Channel Processor Ver2 Module prior to input to this module so that each frame includes RS and virtual channel statistics.

Output Data Format

Output is version 1 packets. Each packet may or may not include the packet header depending on module configuration.

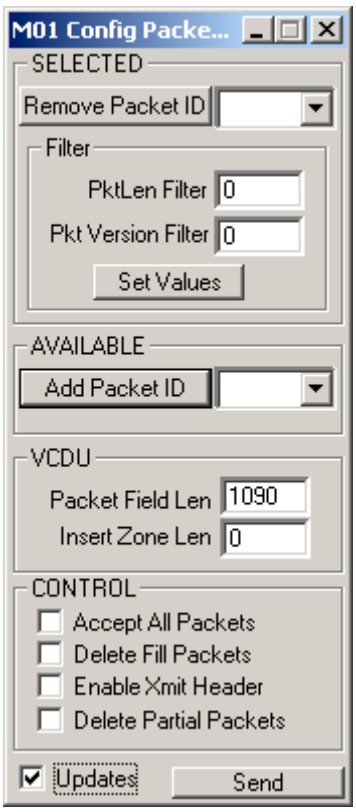
Module Window

When *PacketProcessorVer2.dll* is added to a desktop, the Packet Processor Version 2 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 66-1: PACKET PROCESSOR VERSION 2 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure PACKET PROCESSOR VERSION 2 MODULE Board Parameters sub-menu.



Module 66-2: Config PACKET PROCESSOR VERSION 2 MODULE Window

Module 66-3: PACKET PROCESSOR VERSION 2 MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
SELECTED			
Remove Packet ID	Packet APID values	Any packet entered with "Add Packet ID"	The list available through the down arrow provides APIDs of all packets that will be recognized and output from this module. The "Remove Packet ID" command allows the operator to remove a packet APID from the output packet list. To remove an APID from the list of output packets, select the APID from the list, wait for the selected APID to appear in the window, and then issue the "Remove Packet ID" command. The APID is not removed until the "Remove Packet ID" command is issued. The list of packet APIDs available in this window are those that have been entered with "Add Packet ID." Use this list to view and define filter criteria.
SELECTED/FILTER CRITERIA Packet APIDs selected for re-assembly must also pass filter criteria. Filter criteria are defined on a per APID basis. To view filter criteria for an APID, highlight the APID in the Remove Packet ID window. All Filter criteria that show applies to the highlighted APID only. Filter criteria set to zero means no filtering for that APID. To create filter criteria, highlight the APID in the Remove Packet ID window. Enter the filter criteria, and then issue a "Set Value" command. The filter criteria are then applied to that APID only.			
PktLen Filter	bytes		Defines the acceptable packet length for that APID-- if it doesn't match it is filtered

Parameter	Unit	Range or Values	Description
Pkt Version Filter	Version #	0, 1, or 2 3-bit field	Defines if packet version filtering for that APID. If filtering is enabled, then it defines which packet version is to be processed. Enter the appropriate value to define processing: 1) 0 disables version filtering: no filtering based on packet version 2) 1 means accept and reassemble version 1 packets only 3) 2 means accept and reassemble version 2 packets only The value entered in this field is compared to the actual packet header version field that is read, which is a 3 bit field. The setup value minus one is compared to the header value. Numbers greater than 2 should not be entered since currently there are not packet versions greater than two.
Set Values	command	N/A	Filter criteria is not set or changed until this command is issued. When pressed, this button defines the entered filter criteria for the APID that is showing in the Remove Packet ID window. If filter criteria is changed during a data flow, the filter criteria is not used until the SEND command is also issued.
AVAILABLE			
Add Packet ID	Packet APID values decimal		This field allows the operator to define which packets (by APID) will be reassembled and output from the module. The values set are no relevant if the Accept All Pkts option is enabled. The maximum number of packets that can be defined for re-assembly is 64. To add a packet, enter the APID in the window, and then press the Add Packet ID command. All packet APIDs to be reassembled appear in the Remove Packet ID window. Once entered, a packet APID remains an option in the Add Packet ID window even if it has been removed from the reassembly list. If data is input to the system, the Add Packet ID window reflects all APIDs that are detected in the input data stream (with a maximum of 64). This allows the operator to select APIDs on the fly, however, the change to setup is not implemented until both the Add Packet ID and SEND commands are issued.
VCDU			
Packet Field Len	Bytes	0 to Frame length - (sync, header, CRC, CLCW, insert zone, and RS)	Defines the number of bytes in the input frame that contains packet data: defines the length of the Packet Data Field within the Transfer Frame. The field begins after the first header pointer and continues to either the CRC (if included), CLCW (if included) or Reed-Solomon symbols. Note for reference purposes: version 1 frames have a 10 byte header including sync; version 2 frames have a 12 byte header including sync. Account for an insert zone if present.

Parameter	Unit	Range or Values	Description
Insert Zone Len	bytes	0 to Frame length minus (sync, header, CRC, CLCW, and RS symbols)	Needs to be set up to match the input frame format. If the input frame includes an insert zone, the insert zone length must be defined here. Set to zero for no insert zone. The insert zone precedes the first header pointer.
CONTROL			
Accept All Packets	Selectable Setup State	Enable Disable	When enabled, all input packets are reassembled and output from the module regardless of the packet APID. Checkbox setup: insert a check to enable and remove check to disable.
Delete Fill Packets	Selectable Setup State	Enable Disable	If enabled, then all fill packets are deleted before output. If Accept All Packet is also enabled, Delete Fill Packets is applied after Accept All Packets (all packets accepted except fill). If enabled, then fill packets are filtered prior to output. A fill packet is defined as APID 2047
Delete Partial Packets	Selectable Setup State	Enable Disable	If enabled, then all partial packets are deleted before output. The module determines partial packets by either an incorrect First Header Pointer or VCID sequence error. If disabled, then partial packets are output from the module and a partial packet status flag is set in the module's buffer descriptor (status output buffer).
Enable Xmit header	Selectable Setup	Enable Disable	Defines how reassembled packets are output from the module. When enabled, each packet is output with the packet header still attached. When disabled, the packet header is stripped from each packet prior to output from the module. Checkbox setup: insert a check to enable and remove check to disable. Pkt header is 6 bytes for version 1 packets Pkt header is 14 bytes for version 2, but even if disabled, only 6 bytes are stripped prior to output

Extended Options Window Definition

There is no extended options window for the PACKET PROCESSOR VERSION 2 MODULE.

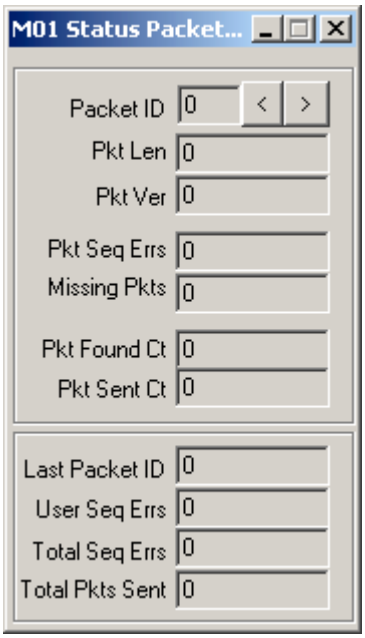
Configuration Notes

The PACKET PROCESSOR VERSION 2 MODULE command window currently does not provide any commanding functions.

Module 66-4: Command PACKET PROCESSOR VERSION 2 MODULE Window

Status Window Definition

When you click the Status button on the PACKET PROCESSOR VERSION 2 MODULE window, a Status PACKET PROCESSOR VERSION 2 window appears, as shown below.



Module 66-5: Status PACKET PROCESSOR VERSION 2 MODULE Window

Module 66-6: PACKET PROCESSOR VERSION 2 MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
APID-SPECIFIC STATUS			
Packet ID	APID	All APIDs detected in the input stream	This field allows the operator to chose which APID statistics will be displayed in the rest of the status window. To choose the APID, use the right/left arrows provided. The available selections are the APIDs defined by the config window for re-assembly. The selection made from this list of packet APIDs defines the APID for which all other APID-specific status in the window applies. To change the display and view status for a different APID, select an APID from the available list-- that APID shows in the window once it is selected. This list reflects the APIDs that have been detected in the input data stream: if an APID is not available in the list, then that means it hasn't detected any packets of that APID.

Parameter	Unit	Range or Values	Description
Pkt Len	bytes	variable	Displays the detected packet length for that APID. Displays the exact information read from the packet header length field. Since the value is read directly from the packet header, the display length is packet length minus one. Displays the Packet length for last packet received with the APID chosen in "Packet ID."
Pkt Ver	Version #	0 or 1	Displays the detected Packet version for last packet received with the APID chosen in "Packet ID." 0=version 1; 1= version 2
Pkt Seq Errs	count	Increments by one for each sequence error	Displays the number of sequence errors that have occurred for the APID chosen in "Packet ID." Reflects the number of times a discontinuity was detected in the packet sequence count read from the packet header.
Missing Pkts	count	Increments by the packet gap size	This is a total number of the missing packets for the APID chosen in "Packet ID." The count is based on packet sequence count errors and the gap in sequence counts. The maximum number of packets that can be detected as missing for each packet gap is a function of the 14-bit sequence count field that is in each packet header.
Pkt Found Ct	count	Increments by one for each packet input	Number of packets found for that APID (all might not have been sent because of the filter criteria).
Pkt Sent Ct	count	Increments by one for each packet output	Number of packets output from the module with the APID chosen in "Packet ID."
MODULE SUMMARY STATUS			
Last Packet ID	APID	11-bit field	Reflects the APID of the last packet input to the module.
User Seq Errs	count	Increments by one for each sequence error	Reports the total number of sequence errors detected in all APIDs defined for reassembly-- in all user selected packets
Total Seq Errs	count	Increments by one for each sequence error	Reports the total number of sequence errors detected in all APIDs including those APIDs not selected for reassembly
Total Pkts sent	count	Increments by one for each packet output	Total number of packets output from the module.

Module 67 PCR-2000 RECEIVER/DEMODULATOR MODULE

MODULE NAME: PCR2000.dll

Key Functions

- Controls Microdyne Telemetry Receiver / Demodulator.
- Reports status from the card.

Required Hardware

- Microdyne PCR-2000 Receiver / Demodulator card.

Add Module Guidelines

One module per receiver card.

Required Input Source, Connections, and Events

INPUT SOURCE: All data is received via cables plugged into the back of the card.

CONNECTIONS: No modules are connected to this module.

EVENTS: No events are generated or expected.

Module Functional Description

The PCR2000 module supports the Microdyne PCR-2000 S-Band Receiver/Demodulator. This module has no data inputs or outputs and cannot be "connected" to any other modules. The Microdyne PCR-2000 is a two-board sandwich, which occupies one ISA slot. It has a dual conversion superheterodyne receiver, and operates in the 2200-2300 MHz (S-Band) range. The Avtec PTP for Windows does not validate this range, since boards with other ranges are available. Care should be exercised when inserting the card set into the computer. Because of the sandwich configuration, a card plugged into the adjacent slot could electrically touch this card. Be sure that there can be no board-to-board contact with other cards in the computer.

Input Data Format

S-Band telemetry.

Output Data Format

Cables at the rear of the board.

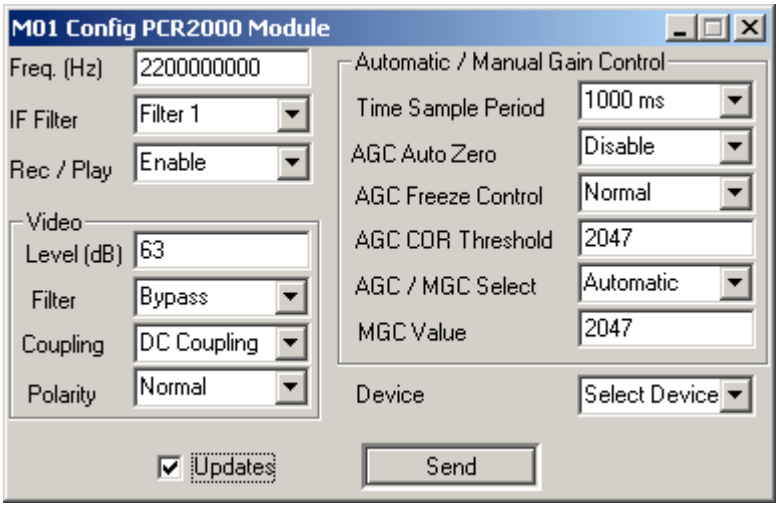
Module Window

When PCR2000.dll is added to a desktop, the PCR2000 dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 67-1: PCR2000 MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure PCR2000 MODULE Board Parameters sub-menu.



Module 67-2: Config PCR2000 MODULE Window

Module 67-3: PCR2000 MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Freq	Hz		Input signal frequency; default is 2200000000 (2.2 GHz)
IF Filter		Filter 1, Filter 2, Filter 3, Filter 4	Select IF filter to use, Filter 1, 2, 3, or 4
Rec/Play		Enable Disable	Enables or Disable data to record line
VIDEO			
Level	dB		Input video signal level

Parameter	Unit	Range or Values	Description
Filter		Bypass Filter 1, Filter 2, Filter 3	Bypass/Enable (Filter 1, 2, or 3) video filter
Coupling		DC Coupling, AC Coupling	DC or AC coupling
Polarity		Normal, Invert	Selects signal polarity (Normal or Inverted)
AUTOMATIC/MANUAL GAIN CONTROL			
Time Sample Period	ms	0.1 ms, 1 ms, 10 ms, 100 ms, 1000 ms,	Selects Time Sample Period (0.1, 1.0, 10, 100, or 1000 ms)
AGC Auto Zero		Disable Activate	Zeroes AGC now. Complete is reported when finished.
AGC Freeze Control		Normal, Freeze	Normal or Freeze
AGC COR Threshold			Input AGC COR Threshold value (0 to 4095)
AGC/MGC Select		Manual, Automatic	Selects Automatic or Manual Gain Control
MGC Value			If MGC enabled, input value between 0 and 4095
OTHER CONFIG			
Device		Select Device	Selects which PCR2000 is being configured

Extended Options Window Definition

There is no extended options window for the PCR2000 MODULE.

Configuration Notes

When you click the Command button on the PCR2000 MODULE window, a Command PCR2000 window appears, as shown below.

Module 67-4: Command PCR2000 MODULE Window

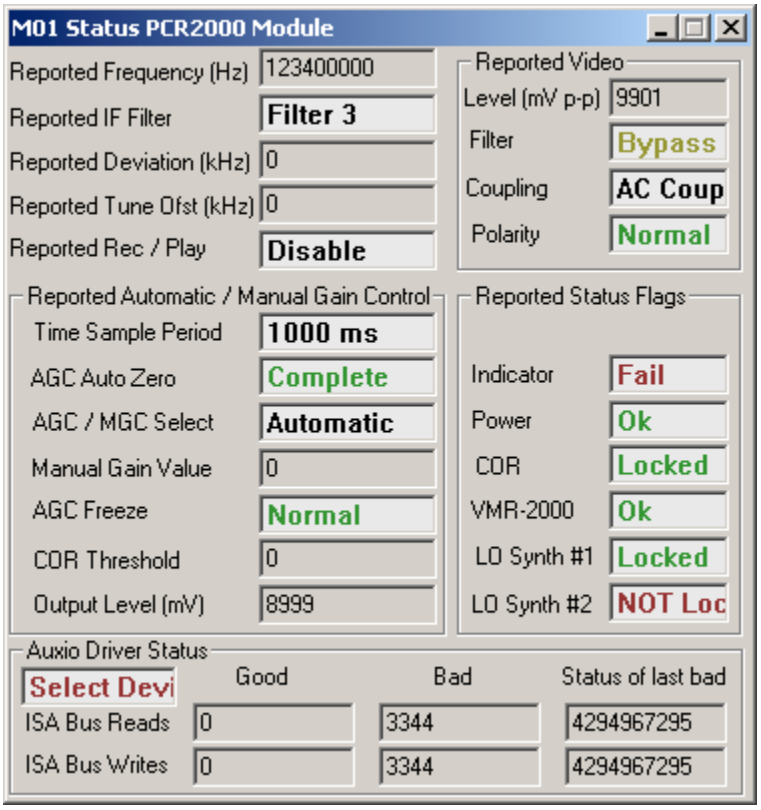
Module 67-5: PCR2000 MODULE Command Window Parameters

Parameter	Unit	Range or Values	Description

Parameter	Unit	Range or Values	Description
No Command Selected			No command
Power On Test			Initiate power on test

Status Window Definition

When you click the Status button on the PCR2000 MODULE window, a Status PCR2000 window appears, as shown below.



Module 67-6: Status PCR2000 MODULE Window

Module 67-7: PCR2000 MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Reported Frequency	Hz	As reported	Displays input frequency
Reported IF Filter		Filter 1 thru 4, as reported	Displays IF filter being used
Reported Deviation	KHz	As reported	Displays deviation from input frequency
Reported Tune Ofst	KHz	As reported	Displays tuning offset

Parameter	Unit	Range or Values	Description
Reported Rec/Play		As reported	Displays if Play/Rec is enabled or disabled
REPORTED AUTOMATIC/MANUAL GAIN CONTROL			
Time Sample Period		As reported	Displays time sample period
AGC Auto Zero		As reported	Displays status of AGC Auto Zero
AGC/MGC Select		As reported	Displays if automatic or manual gain control is being used
Manual Gain Value		As reported	Displays manual gain value
AGC Freeze		As reported	Displays status of AGC freeze
COR Threshold		As reported	Displays COR threshold value
Output Level	mV	As reported	Displays output signal level
REPORTED VIDEO			
Level	MV p-p	As reported	Displays video level
Filter		As reported	Displays filter in use
Coupling		As reported	Displays AC or DC coupling in use
Polarity		As reported	Displays video polarity
REPORTED STATUS FLAGS			
Indicator		OK / Fail, as reported	Displays general board status
Power		As reported	Displays Power status
COR		As reported	Displays COR (Command Output Ready) status (Locked or Not Locked)
VMR-2000		As reported	Displays VMR-2000 status (OK/Fail)
L0 Synth #1		As reported	Displays status of LO Synth #1 (Locked or Not Locked)
L0 Synth #2		As reported	Displays status of LO Synth #2 (Locked or Not Locked)
AUXIO DRIVER STATUS			
ISA Bus Reads		Good, Bad, Status of Last Bad	Displays number of good and bad ISA Bus Reads
ISA Bus Writes		Good, Bad, Status of Last Bad	Displays number of good and bad ISA Bus Writes

Module 68 PSK SUMMER MODULE

MODULE NAME: PSKSummer.dll

Key Functions

- Electronically sums PSK, TTL, Bipolar signals and outputs a user selectable combination of signals
- Controls ATSC PSK Summer Card

Required Hardware

- The following board must be installed in the Avtec PTP for Windows system:
  - ✓ ATSC PSK Summer card
- Cables to provide signals

Add Module Guidelines

Only one ATSC PSK Summer board can be installed in a computer. Only one CMOD software module should be loaded to control it.

Required Input Source, Connections, and Events

INPUT SOURCE: The module receives all data and signals from external cables. These signals include PSK, TTL, and BiPolar.

CONNECTIONS: No module connections should be made to or from this module. No data is received or sent via the Avtec PTP for Windows.

EVENTS: No events are expected or generated by this module.

Module Functional Description

This module electronically sums the three inputs. An option is provided to exclude any of the signals from the output. There is a PSK output, and there are two independently buffered summed outputs.

Input Data Format

Electronic signals in the prescribed waveform.

Output Data Format

PSK and PSK summed (× 2).

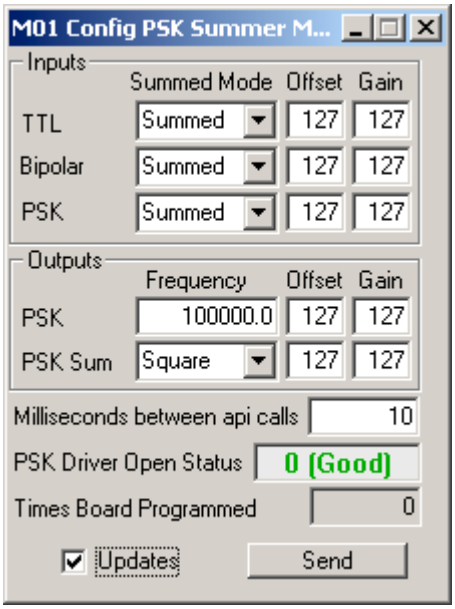
Module Window

When PSKSummer.dll is added to a desktop, the PSK Summer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 68-1: PSK SUMMER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure PSK SUMMER MODULE Board Parameters sub-menu.



Module 68-2: Config PSK SUMMER MODULE Window

Module 68-3: PSK SUMMER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
INPUTS			
TTL Setup			
TTL Summed Mode	Selectable Setup State	Summed Not	Selects whether or not the TTL input should be summed in the output.
TTL Offset		0 to 255	Offset to be applied to the TTL signal. (8bit potentiometer)
TTL Gain		0 to 255	Gain to be applied to the TTL signal. (8bit potentiometer)

Parameter	Unit	Range or Values	Description
Bipolar Setup			
Bipolar Summed Mode	Selectable Setup State	Summed Not	Selects whether or not the Bipolar input should be summed in the output.
Bipolar Offset		0 to 255	Offset to be applied to the Bipolar signal. (8bit potentiometer)
Bipolar Gain		0 to 255	Gain to be applied to the Bipolar signal. (8bit potentiometer)
PSK Setup			
PSK Summed Mode	Selectable Setup State	Summed Not	Selects whether or not the PSK input should be summed in the output.
PSK Offset		0 to 255	Offset to be applied to the PSK signal. (8bit potentiometer)
PSK Gain		0 to 255	Gain to be applied to the PSK signal. (8bit potentiometer)
OUTPUTS			
PSK Setup			
PSK Frequency		1000.0 to 2500000.0	PSK output frequency
PSK Offset		0 to 255	Offset to be applied to the PSK signal. (8bit potentiometer)
PSK Gain		0 to 255	Gain to be applied to the PSK signal. (8bit potentiometer)
PSK Sum Setup			
PSK Wavetype		Square Sine	Selects sine or square wave output
PSK Sum Offset			Offset to be applied to the PSK signal. (8bit potentiometer)
PSK Sum Gain			Gain to be applied to the PSK signal. (8bit potentiometer)
BOARD-LEVEL SETUP			
Milliseconds between api calls			Time between calls to the PSK Summer driver. This parameter should be tweaked based on the speed of the host computer.
PSK Driver Open Status			Status returned from the open call to the PSK Summer driver
Time Board Programmed			Count of how many times the user has reset and programmed the board. Note that there is no success/failure status reported by the driver.

Extended Options Window Definition

There is no extended options window for the PSK SUMMER MODULE.

Configuration Notes

Note that there is no status reporting from the card. Only the open call to the software driver reports status.

Status Window Definition

There is no status window for the PSK SUMMER MODULE.

Module 69 RANDOMIZER MODULE

MODULE NAME: Randomizer.dll

Key Functions

- Provides CCSDS-defined randomization or de-randomization of input data unit
- Provides user-configurable offset to start and end of randomization

Required Hardware

- No special hardware requirements.

Add Module Guidelines

Any number of RANDOMIZER MODULES can be added to a desktop: do not exceed 32 modules/desktop limit.

Required Input Source, Connections, and Events

INPUT SOURCE: Receives data input from any other module.

CONNECTIONS: Outputs to port one, which must be connected to another module's input or the data is lost.

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The RANDOMIZER MODULE randomizes OR de-randomizes input data. The module XORs a pseudo-randomized pattern onto each input data unit. If the input data is already encoded, the result is de-randomized data output. If the input data is NOT randomized, then the output is randomized data units.

The module uses the CCSDS randomization definition provided in CCSDS 201.0-b-2 paragraph 3.3.1.1. The CCSDS-defined, pseudo-random pattern follows:

$H(x) = x^8 + x^6 + x^4 + x^3 + x^2 + x + 1$

To provide versatility, the RANDOMIZER MODULE allows the user to define the start and end of the randomization zone relative to the input data unit's start. The module provides a user-configurable offset and length from data unit start to define the randomization zone. The offset and length is fixed for the data flow.

Input Data Format

Any data can be input. Maximum Avtec PTP for Windows buffer size is 16 Kbytes and this is the input size limitation for this module.

Output Data Format

Output data format is dependent on the input. Output is either randomized or de-randomized data units of the same length input.

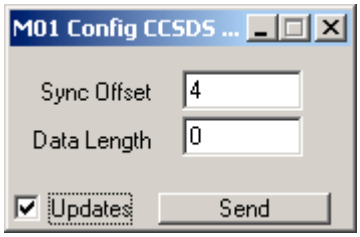
Module Window

When Randomizer.dll is added to a desktop, the Randomizer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters.

Module 69-1: RANDOMIZER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure RANDOMIZER MODULE Board Parameters sub-menu.



Module 69-2: Config RANDOMIZER MODULE Window

Module 69-3: RANDOMIZER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Sync Offset	bytes	0 to frame length	Allows the operator to define a number of bytes at the start of the data unit that is not encoded/decoded. Defines the number of bytes to skip before beginning the XOR.

Parameter	Unit	Range or Values	Description
Data Length	bytes	16 K	Allows the operator to define a number of bytes at the data unit end that is not encoded/decoded. A zero value means encode/decode to frame end. Any other value defines a fixed length that is encoded/decode regardless of input frame length. The value entered here is relative to sync offset. It tells the module, starting at the offset position, how many bytes to encode/decode. If left to zero, it will encode/decode until the end of input frame When configuring, remember to account for RS symbols when setting the length if the RS was applied after randomization

Extended Options Window Definition

There is no extended options window for the RANDOMIZER MODULE.

Configuration Notes

There are no configuration notes for the RANDOMIZER MODULE.

Status Window Definition

There is no status window for the RANDOMIZER MODULE.

Module 70 RAW SOCKETS MODULE

MODULE NAME: RawSockets.dll

Key Functions

- Provides bi-directional, connectionless socket
- Optionally strips IP header from socket input prior to output to port 1
- Appends IP header to each buffer output to socket

Required Hardware

- No special hardware required.

Add Module Guidelines

Any number of RAW SOCKETS MODULES can be added to a desktop as long as the 32 modules per desktop limit is not exceeded. It always good practice to monitor CPU usage at high data rates and for complicated desktops.

Required Input Source, Connections, and Events

INPUT SOURCE: Can either receive input from a socket, any other module, or both simultaneously.

CONNECTIONS: Outputs to port one when receiving from a socket. Outputs to socket when receiving from another module.

EVENTS: Neither generates an event or requires an event input.

Module Functional Description

The RAW SOCKETS MODULE is a bi-directional, connectionless, socket. The module transmits anything received on its input port to the socket, and it outputs to port 1 anything received on the socket. Can provide simultaneous, bi-directional transfers.

The RAW SOCKETS MODULE inserts an IP header on each data buffer output to the socket. Depending on setup, the module either strips the IP Header (20 bytes) from each data unit received from the socket prior to output to port one, or it transfers each data unit as received to port one (IP header is NOT removed).

This socket connection requires no acknowledgements. By using only the IP Header, it reduces overhead and traffic.

Input Data Format

Expects buffer input from another module for output to socket. Depending on the source module, the buffer may be on a frame or packet boundary. Socket input is a data unit that may or may not include an IP header.

Output Data Format

This module appends an IP header (20 byte header) to each input buffer before transfer to socket.

For data received from the socket, the module will strip the IP header if set to strip IP header. Otherwise, it outputs the data untouched (it does not remove the IP header).

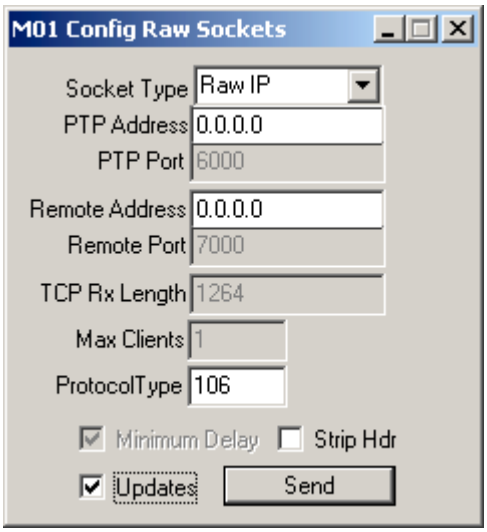
Module Window

When RawSockets.dll is added to a desktop, the Raw Sockets dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 70-1: RAW SOCKETS MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure RAW SOCKETS MODULE Board Parameters sub-menu.



Module 70-2: Config RAW SOCKETS MODULE Window

Module 70-3: RAW SOCKETS MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Socket Type	Selectable Setup	Raw IP, Reliable Delivery	Defines the type of raw socket that is to be used: 1) Raw IP: the raw IP setup that is most commonly used 2) Reliable Delivery is a type of raw socket. For definition refer to window's protocol.
PTP Address	address	N/A	Defines local Avtec PTP for Windows address- the address of the machine on which the module actually resides. Applies to multi-homed machines-- machines with multiple network boards installed. This address defines the Class C address of the network board that you want to use. An all zero entry means use the first board found. IP address (0.0.0.0 specifies the default machine IP address)
PTP Port	N/A	N/A	This configuration parameter does not apply to raw sockets.
Remote Address	address	N/A	Defines the remote (destination) address that is put into the IP header. Defines the class C address of the remote machine to which the module is sending data. Can also enter a class D address. Zero entries are illegal.
Remote Port	N/A	N/A	This configuration parameter does not apply to raw sockets.
TCP Rx Length	N/A	N/A	This configuration parameter does not apply to raw sockets.
Max Clients	N/A	N/A	This configuration parameter does not apply to raw sockets.
Protocol Type	decimal		Module puts this value into the IP header. The value does not affect raw sockets processing. Defines the protocol that sits on top of the raw IP layer. Is user-defined and should be matched at both send and receive sides. Receive side may use this value to filter.
Minimum Delay	N/A	N/A	This configuration parameter does not apply to raw sockets.
Strip Hdr	Selectable Setup	Enable Disable	When enabled, it applies to socket data input. If enabled, the module strips the IP header prior to output on port one.

Extended Options Window Definition

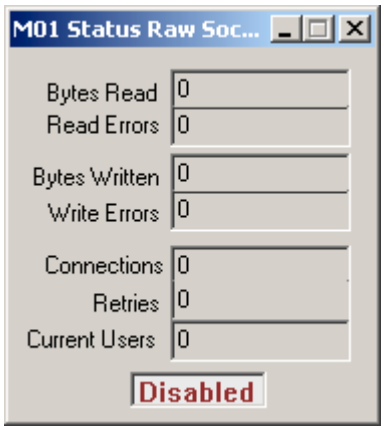
There is no extended options window for the RAW SOCKETS MODULE.

Configuration Notes

There are no configuration notes for the RAW SOCKETS MODULE.

Status Window Definition

When you click the Status button on the RAW SOCKETS MODULE window, a Status RAW SOCKETS window appears, as shown below.



Module 70-4: Status RAW SOCKETS MODULE Window

Module 70-5: RAW SOCKETS MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Bytes Read	Bytes	Incrementing counter	Total bytes received on the socket. Count of bytes read
Read Errors	Errors	Incrementing counter	Count of read errors. For raw sockets, this indicates an anomaly in protocol stack
Bytes Written	Bytes	Incrementing counter	Total number of bytes transmitted to the socket. Count of bytes written
Write Errors	Errors	Incrementing counter	Count of write errors. For raw sockets, this indicates an anomaly in protocol stack-- for example, can't find router.
Connections	Connections	Incrementing counter	Current number of connections. Number of times a successful connection has been made includes re-connections.
Retries	Retries	Incrementing counter	Number of connection retries-- number of times a connection was attempted
Current User	users	Count	Number of current users.
Status Window	N/A	Disabled, Connected	Reports socket status. As soon as module is enabled it should say connected unless it can't find the local router or the local Avtec PTP for Windows address cannot be found.

Module 71 SOFTWARE SYNCHRONIZER MODULE

MODULE NAME: SoftSync.dll

Key Functions

- Bit synchronizes serial telemetry streams
- Variable length sync pattern delimited frames
- MicroSecond timetag adjustment for any selected bit in frame
- Fixed Length Frames
- Variable length sync and tail pattern delimited frames

Required Hardware

- This module does not require any special hardware to be installed in the Avtec PTP for Windows system.

Add Module Guidelines

One instance of this module per stream to be synchronized into frames can be added to a desktop. Each instance is independent.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives data from other modules. Only one stream can be processed at one time.

CONNECTIONS: An input, such as a simulator, serial input, or any other module which sends data which need to be synchronized and separated into frames.

EVENTS: No events are expected and none are generated.

Module Functional Description

This module accepts data from other modules up to 8192 bytes per buffer. The data is inserted into a reserved circular buffer sized at 65536 bytes. Bits are inserted into a shift register, one at a time, until a user defined pattern is found to achieve lock. An inverted pattern may also be accepted, if desired. Once lock is detected, the bits are copied into a new buffer to be output as a frame. The frame may be delimited by a fixed number of bytes, the start of the next sync pattern, or by a user defined tail pattern. The circular buffer is not allowed to run dry. The maximum expected frame length plus 40 bits of data will be retained until more data arrives or a timeout condition exists. The buffer is completely flushed after a user-defined period of no data occurs.

The data rate is calculated using the buffer time attached to each input buffer. The number of bits in each buffer and the difference in microseconds between successive buffers allow the

bit-rate to be calculated, in nanoseconds. This provides the actual time per bit to be multiplied by the bit offset in the original input buffer. This value is added to the microsecond-resolved time attached to that original input buffer. The use of nanoseconds-per-bit serves as guard to protect the microsecond buffer time from accumulating error. The user can select which bit in the frame is to be time tagged.

Input Data Format

The entire input buffer is data to be synchronized. No fields exist.

Output Data Format

The output buffers are frames, beginning with the synchronization pattern and extending contiguously to the end of one frame. The frame size can be fixed or variable, with user defined limits.

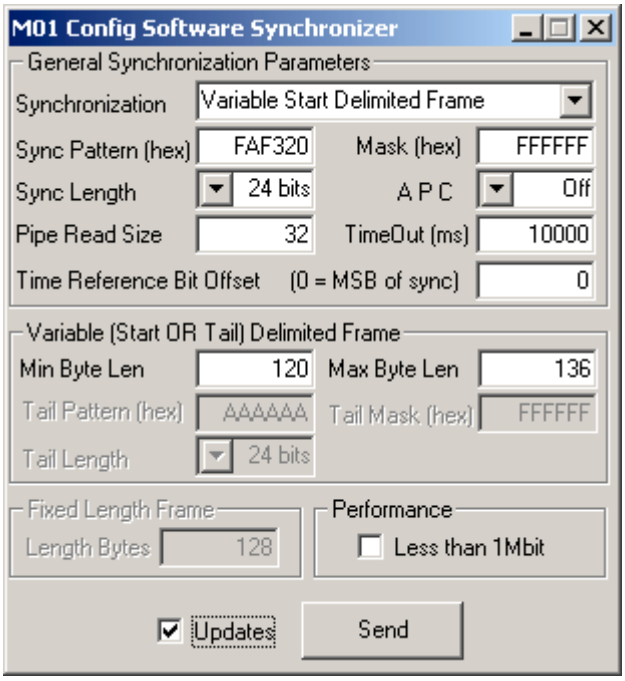
Module Window

When SoftSync.dll is added to a desktop, the Software Synchronizer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 71-1: SOFTWARE SYNCHRONIZER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown in the SOFTWARE SYNCHRONIZER MODULE Board Parameters sub-menu.



Module 71-2: Config SOFTWARE SYNCHRONIZER MODULE Window

Module 71-3: SOFTWARE SYNCHRONIZER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
GENERAL SYNCHRONIZATION PARAMETERS			
Synchronization	Selectable Setup State	Variable Start Delimited Frame, Variable Tail Delimited Frame, Fixed Length Frame	Selects the type of synchronization to be done. Frames begin with the sync pattern. Variable Start Delimited identifies frames to end with the bit just prior to the next sync pattern. Variable Tail Delimited causes frames to end with the last bit found in a tail pattern; bit between the tail pattern and the next sync pattern are discarded. Fixed Length causes the frame to be a specified number of bytes long. Data occurring after the frame and before the next sync pattern is discarded.
Sync Pattern (hex)	Hexa-decimal	16 bit, 24 bit or 32 bit pattern	Specifies the sync pattern. If the pattern is 16 bits or less, enter the four hexadecimal digits. If the pattern is 17 to 24 bits, enter the six hexadecimal digits. For a pattern 25 to 32 bits, enter the eight hexadecimal digits. The most significant bits in the pattern identify the most significant bits to be placed in the frame.
Mask (hex)	Hexa-decimal	16 bit, 24 bit or 32 bit pattern	Specifies the mask for the sync pattern. See above.
Sync Length	Selectable Setup State	16 bits, 24 bits or 32 bits	Used by the software to shift the pattern into the appropriate position in a 32 bit word.

Parameter	Unit	Range or Values	Description
APC	Selectable Setup State	Off, On	Automatic Polarity Correction. In the Off position, only the sync pattern is recognized as sync. In the On position, inverted sync will also be recognized as sync. If the sync is inverted, the data will be inverted automatically by the software to correct the polarity.
Pipe Read Size	Bytes	Maximum frame size or greater	Allows control over how large the circular buffer must grow before frames will be extracted using the frame synchronization logic. A larger value, say three times the frame size will causes the data to be bursted out in 3 frame chunks, but will improve the throughput of the system in general. If Pipe Read Size is less that the maximum frame size, it will internally be forced to the frame size.
TimeOut (ms)	Milli-seconds	0 to 120000	Defines when the circular buffer is be flushed of remaining frames. If a new buffer has not been received within (ms) milliseconds, the buffer will be flushed, and the synchronizer will be forced into the Search (re-synchronize) mode.
Time Reference Bit Offset	Zero offset number	0 to frame size in bits	Specifies which bit in the frame is to be time tagged. Zero indicates the most significant bit.
VARIABLE (START OR TAIL) DELIMITED FRAME			
Min Byte Len		10 to 1024	Specifies the minimum frame size in bytes (for variable length frames only)
Max Byte Len		Min+1 to 1024	Specifies the maximum frame size in bytes (for variable length frames only)
Tail Pattern (hex)	Hexa-decimal	16 bit, 24 bit or 32 bit pattern	Specifies the tail pattern which indicates the end of a frame. See Sync Pattern, above. This item applies to tail delimited sync function only.
Tail Mask (hex)	Hexa-decimal	16 bit, 24 bit or 32 bit pattern	Specifies the mask for the tail pattern. See above. This item applies to the tail delimited sync function only.
Tail Length	Selectable Setup State	16 bit, 24 bit or 32 bit	Used by the software to shift the pattern into the appropriate position in a 32 bit word.
FIXED LENGTH FRAME			
Length Bytes		10 to 1024	Specifies the length of a fixed length frame.
PERFORMANCE			
Less than 1MBit			

Extended Options Window Definition

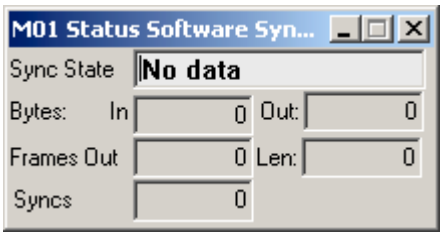
There is no extended options window for the SOFTWARE SYNCHRONIZER MODULE.

Configuration Notes

Parameters such as Pipe Read Size need to be configured by trial and error. A larger value will cause several frames to be burst out back to back, then a longer space between frames. A large number will increase the throughput of the system in general, a lower number, down to the frame size will cause less buffers to be held up, decreasing jitter. This parameter needs to be adjusted to fit the requirements of the intended use of this module.

Status Window Definition

When you click the Status button on the SOFTWARE SYNCHRONIZER MODULE window, a Status SOFTWARE SYNCHRONIZER window appears, as shown below.



Module 71-4: Status SOFTWARE SYNCHRONIZER MODULE Window

Module 71-5: SOFTWARE SYNCHRONIZER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Sync State	Discrete State	No data, Search, Lock Normal, Lock Inverted	State of the synchronizer logic. Note that there is no flywheel state. Search is red in color, Lock Normal and Lock Inverted are green, and No data is black.
Bytes In	bytes	Total count	Count of bytes received by the module.
Bytes Out	bytes	Total count	Count of bytes output by the module.
Frames Out	frames	Total count	Count of frames output by the module.
Frame Len	bytes	Reflects processing	Byte length of the last frame output.
Syncs	Sync Patterns	Total count	Count of frame syncs recognized.

Module 72 STRIPPER MODULE

MODULE NAME: Stripper.dll

Key Functions

- Allows user to modify a data stream by moving any size pieces of data to different offsets within the buffer, or eliminating them entirely.

Required Hardware

- None.

Add Module Guidelines

There is no limit as to module iteration quantity.

Required Input Source, Connections, and Events

INPUT SOURCE: Data from other modules that send data.

CONNECTIONS: There must be at least one input connection and a least one output connection. Up to four output ports can be used: port one through port four.

EVENTS: No events are expected or generated.

Module Functional Description

This module accepts data from other modules. Pieces of the buffer are transferred to a new buffer for output. Unspecified areas of the output buffer are filled with binary zeroes. Parts of the input buffer that are not moved are lost. Different output buffer contents may be derived from one input buffer and output on different ports. This allows up to four user-defined stripped streams to be made up from one copy of the data.

The user defines a list of up to 400 components (or items). These items contain instructions used at run time to extract sections of data from the input buffer and transfer them to the output buffer. Each component or item in the list consists of a source byte number, the length, and a destination byte number to transfer. Fill patterns may be created and inserted instead of real data. The buffer previous to the current input buffer is retained in the case that data is required from that buffer also. Only one previous buffer is retained.

Not all of the 400 items in the list need to be executed for each input buffer. The user may define items that execute unconditionally or conditionally. If "No Counter" is selected, ALL defined items execute unconditionally for each input buffer. If a counter is defined, either 8 bits or less, or 9 to 16 bits, the items will be executed conditionally, according to the

following conditions. First, the value of the counter is extracted from the input. The counter doesn't really have to be a counter, it could be any desired data point that means the users' requirements to switch upon. The specified mask is applied to the value. The specified bit shift is executed upon the value. If the resultant value is equal to the counter value entered in the gui for the expected counter value, the extraction defined in this component is executed. If it is not, a check is made to see if the counter is a defined subcom of the current counter value. For example, if the subcom parameter is set to 4, every fourth counter value greater than the expected counter value will cause the extraction component to execute also. If the expected counter value is 0, and the subcom is 4, then the component will execute when the counter data value is 0, 4, 8, 12, 16, … and so forth. Secondly, if the expected counter is 10, and the subcom is 3, the extraction component will execute when the data value of the counter is 10, 13, 16, 19, 22, …and so forth. A value if 1 for the subcom will cause the extraction component to execute at every data value of counter greater than or equal to the specified expected counter value. A value of zero disables the subcom feature; the conditional extraction will occur only when the counter data value is exactly equal to the expected counter value. A value of 0 for the expected counter value coupled with a subcom value of 1 creates an unconditional entry when a counter is defined.

Input Data Format

There is no specific format. The buffer may contain any type of data.

Output Data Format

Completely user definable. Each portion of the output buffer is defined by the console gui or by the Remote Interface Library (RIL) function calls. The length of the output buffer is determined automatically by the last data inserted into the output buffer. This means the output data length could vary if the user defines conditional components at the end of the buffer. If the user wants to guarantee a specific data length, an unconditional extraction definition should be made to output whatever parameter is located at the end of the buffer.

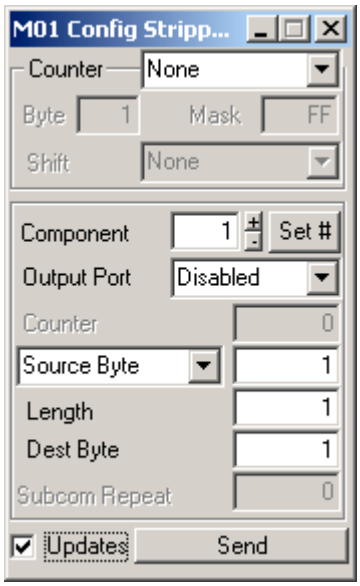
Module Window

When Stripper.dll is added to a desktop, the Stripper dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 72-1: STRIPPER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure STRIPPER MODULE Board Parameters sub-menu.



Module 72-2: Config STRIPPER MODULE Window

Module 72-3: STRIPPER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
COUNTER			
Counter		None, Up to 8 bits, 9 to 16 bits	Defines a frame counter or Virtual Channel ID. Could also be any other value within the data that the user wants to use as basis for determining what parts of the input buffer to put somewhere in the output buffer. Defines the size of the counter, up to 8 bits or 9 to 16 bits, or that no counter is being used for buffer selection.
Byte			Counter byte offset from start of buffer. This is byte number starting with 1 for the most significant byte.
Mask			Counter byte mask. This mask is applied to the counter.
Shift		None, Left 1 to 7 bits, Right 1 to 7 bits	Counter shift action. This action is applied to the counter.--
COMPONENT			
Component	number	1 to 400	Specifies the component number to be defined in the remaining window boxes. A list of components is made, starting with number 1, stipulating the actions to be taken by the stripper. Small + and – buttons allow the user to step thru the list easily, or enter an item number and push Set# button.

Parameter	Unit	Range or Values	Description
Set #	Button	1 to 400 in the Component box	Sets a new component number to goto in the list. After entering the number in the component number box, push the Set# button to go directly to that item in the list, or use the + and – buttons to increment or decrement thru the list.
Output Port		Disabled, Port 1, Port 2, Port 3, Port 4	Port that this segment is to be output.
Counter		Value	Expected Counter Value. If the counter extracted from the data equals the value entered here, the extraction is preformed. If No counter was selected above, the extraction is preformed unconditionally, regardless of value entered.
Source Byte		Input buffer extraction start point	Beginning of extraction point for this component (n of 400). This byte number begins with 1 for the most significant byte. Only byte boundary extractions can be defined. The length of the extraction is defined below, Length.
Hex Byte Fill			Alternate option to Source byte. Allows the user to insert a fill pattern or sync pattern (or anything else) into the data. The data to be inserted is specified in hexadecimal in the box rather that coming from the input data.
Prev Frame Byte			Alternate option to Source byte. Allows the user to insert data from the buffer which came into this module previously. This allows old data to be combined into the output with new data as desired.
Length			Length of the Transfer. This parameter defined the byte length of the extraction defined by the Source Byte parameter, or the length of the fill data inserted by the Hex Byte Fill parameter, or the length of the data coming from the previous buffer.
Dest Byte			Start byte in the output buffer. This number is relative to 1.
Subcom Repeat			Used with the Counter value to allow a subcom repeat function. The extraction can be made conditional on the extracted data value (counter) being equal to the value defined in the Counter Value above, or any subcom, where subcom is defined here to mean every n^{th} repetition of the value defined here. For example, 4 would indicate every fourth frame after the Counter Value. If Counter Value was 0, then the extraction would occur at frame Count 0, 4, 8, 12, 16,…and so forth. The value 1 indicates every frame including and after the Counter specified. The value 0 indicates NO subcom; extraction will be preformed when the Counter value is equal to the extracted data, and at NO other time.

Extended Options Window Definition

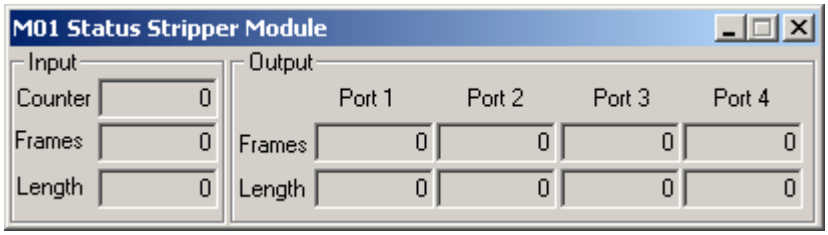
There is no extended options window for the STRIPPER MODULE.

Configuration Notes

There are no configuration notes for the STRIPPER MODULE.

Status Window Definition

When you click the Status button on the STRIPPER MODULE window, a Status STRIPPER window appears, as shown below.



Module 72-4: Status STRIPPER MODULE Window

Module 72-5: STRIPPER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
INPUT			
Counter			Value of Counter in the data
Frames			Number of buffers input
Length			Last buffer input length
OUTPUT			
Frames			Count of buffers / frames output for each port, 1 thru 4
Length			Length of last buffer output from each port, 1 thru 4

Module 73 TDM SIMULATOR MODULE

MODULE NAME: Tdmsim.dll

Key Functions

- Simulates telemetry at a rate defined by the event input rate: module sending events to this module controls data output rate.
- User defined frame sync parameters, identification counters, patterns, fill data capabilities.

Required Hardware

- Since this module expects event input, serial output module hardware should be used to generate a reliable telemetry rate. If no hardware is available, a module that sends events is acceptable as input to this module. The CPU Timer module is an example of a module that sends events at a specified rate, subject to computer resource availability.

Add Module Guidelines

No module iteration restrictions. Each instance of this module generates one output stream.

Required Input Source, Connections, and Events

INPUT SOURCE: This module can received data buffers from other modules and interleave the input data into user defined fill areas of the generated data.

CONNECTIONS: If interleaving is in use, an input connection to this module from another module is required. If no interleaving is required, no data buffer input to this module is necessary. There must be an output connection from port 1 of this module to another module to output data.

EVENTS: Event inputs are required. One event input causes one simulated data buffer to be output. The output data rate is therefore defined by another module that is sending the events to the TDM Sim module.

Module Functional Description

The TDM Simulator Module (Time Division Multiplex Simulator) simulates a user-defined TDM data stream. It is an event-driven module, and therefore requires a CPU Time Module or Serial Output Module to generate an event for each frame of data. The user can configure multiple ID counters and Fill Patterns for the simulated stream. A CRC codeword can also be added to the stream. Optionally, the user may elect to interleave streams from other modules. The TDM Sim module does not protect against interleave overruns or underruns, since it is required to output an entire frame when the event arrives.

Input Data Format

No specific input data format; data buffers are copied into the interleave buffer for the sending module as received. Buffers with odd lengths are adjusted to an even 16 bit boundary. Shuttle interleave data is prepended with a PB1 time stamp and the user provided Shuttle ID.

Output Data Format

The user defines the output data format using the Config window or by Remote Interface Library (RIL) function calls.

The user defines the position of the synchronization pattern. The sync pattern may appear either at the beginning or end of the frame. The sync pattern is defined to be either 24 or 32 bits, based on the value entered for the sync pattern. To define a 24 bit sync, enter the pattern containing six significant hexadecimal digits. In this case, only 3 bytes would be loaded into the output buffer to be sent. If the value in the sync pattern entry field numerically exceeds hexadecimal FFFFFF, the software uses a 32 bit sync field. A leading zero would be prepended if only seven hexadecimal digits are entered; for example, the entry 1B2C3D4 would result in the pattern 01B2C3D4 being used. The user cannot define a 32 bit pattern which begins with eight or more binary zeroes; the module would read this as a 24 bit pattern beginning with the ninth bit. If a 32 bit sync pattern containing zeroes in the most significant eight (or more) positions, it is suggested that the user put the sync pattern last and define the previous byte to the value zero. Another option is to run the output data thru the Scatter Gather module or the Stripper module to provide further, more advanced methods of data manipulation.

The actual frame length of the data is displayed in the Output Frame Sync section of the gui. It is important to understand that the TDM Sim module does not control the length of the frame, it is defined by the module sending the event to TDM Sim. All valid Avtec PTP for Windows events contain a buffer length. See the CPU Timer module or the Serial Output module to enter the desired data length. Any user data defined to be located physically after the buffer length defined by the module sending the events will be lost.

The TDM Sim output buffer is always loaded in a prescribed order. First the entire buffer is set to binary zeroes. Next, the user defined fill data entries are loaded into the buffer, one at a time, starting with the numerically first entry which is enabled. The ID Counters are then loaded into the buffer, starting with the numerically first enabled counter definition. The synchronization pattern is then entered into the data, if it has been enabled. Lastly, if CRC processing is enabled, these two bytes will be written to the buffer. The buffer is then posted. This exact sequence is useful to know, since it is possible to overwrite one or more fields in the data buffer with the next definition prior to the buffer being sent. The use of Shuttle Interleave XOR function depends on setting the buffer to a pattern and then overwriting it.

The user may define up to 30 counters to be inserted into the data. The entry field at the top of the ID Counter section defines which of the counter is to be acted upon when the "Send"

button is pressed. Only one counter can be defined at a time, the next one can be defined once the "Send" button is pressed. The user specifies whether the specific counter increments, decrements, or remains static. This attribute can be changed while data is flowing, to induce sequence errors. The start byte location, relative to 1, is entered, followed by the step size of the counter, and the minimum and maximum values of the counter. If the maximum value of the counter is defined to be 255 or less, one byte is loaded into the TDM Sim output buffer. If the maximum value of the counter is defined to be 256 or greater, two bytes will always be loaded into the TDM Sim output buffer, even if the value can be held in one byte. The value will be loaded as a big endian value, with the eight most significant bits located at the byte position specified by Start Byte, and the least significant eight bits located at Start Byte + 1. The maximum value of the Maximum Value field is 65535; if the user exceeds this value, only the sixteen least significant bits of the entry will be accepted.

The user may define up to 30 fill fields to be inserted into the data. The entry field at the top of the Fill Data section defines which of the fill definitions is to be acted upon when the "Send" button is pressed. Only one fill definition can be defined at a time, the next one can be defined once the "Send" button is pressed. The user specifies the start byte and field size.

If Dynamic fill data is chosen, the TDM Sim buffer fill field changes dynamically. The value to insert into each byte of a dynamic fill field is defined as the byte offset into the individual fill field, relative to 0, plus the value of the first byte of the fill field plus one. For example, assume a fill field four bytes long; the initial buffer sent would contain hexadecimal 00 01 02 03 in the fill field. The second frame will contain 01020304 in the fill field. The third frame send would contain 02 30 04 05 in the fill field. When a byte reaches the value FF, it wraps to 00, without affecting other bytes in the fill field. All fill fields are independent.

Byte Position fill data loads each byte of the field with its' byte offset (position), relative to zero, as the value of the byte. For example, byte 10, counting from to 1 at the start of the buffer, will contain a hexadecimal 09. Byte 11 will contain a hexadecimal 0A. Each frame will contain the same data in the Byte Position fill option. Only the eight least significant bits of the position are used, the value rolls over to 00 after each occurrence of FF.

ID Counter fill data loads the eight least significant bits of the value of the correspondingly defined ID Counter into each byte in the fill field. For the purposes of the ID Counter fill field, fill field definition number 1 corresponds with the counter defined in the ID Counter section, definition number 1. For example, assume fill field number 6 is set to ID Counter. In this case, the least significant eight bits of counter number 6 defined in the ID Counter section of the gui would be used to fill each byte in the fill field.

User specified fill data loads the hexadecimal value specified in the in the User Specified field into each byte of the fill field.

Interleaving loads the selected Fill Field with the specified number of bytes from the data input to this module from the corresponding module number. The Fill Field number corresponds to the module number from which the data comes. Each Fill Field set for interleaving has its' own 65536 buffer containing an "in" and "out" pointer. When a buffer is

received from another module, the data is copied into the interleave buffer allocated for that specific module. Input buffers with odd lengths are automatically extended to an even byte boundary by appending one byte containing hexadecimal FF. If Shuttle Interleaving was selected, the data in the buffer is prepended with the PB1 time of obtained from the sending buffer attached PB4 time stamp before put into the interleave buffer for that module. The user is responsible for metering the data into the TDM Sim module at the same effective bit rate the data is being interleaved into the output fill fields. This metering must take into account the size of the interleave field and the output rate of the TDM Sim module. The 65536 byte buffer allows for some error in the metering algorithm, dependent on bit rate. If the buffer is allowed to underrun, each byte in the TDM Sim output will be set the User Specified hexadecimal byte value, since no data is available. If the user overruns the buffer, the "in" pointer will wrap to the beginning and pass the "out" pointer, overwriting older data before it could be interleaved. The overrun errors shows up as sequence errors in the users TDM Sim output data. Optionally, the user may exclusive-or (XOR) the interleave data into the fill field. This is the Interleave Mode switch. If set to XOR, the interleave data is exclusive-ored with the data previously existing in the buffer. To user this feature, the user first loads the TDM Sim output buffer with a specific value(s), using the Fill Data will a User Specified pattern(s). This fill data would be entered in a Fill Data entry number less than the module number sending the data, since the entries in the Fill Data are processed in order. The user would then define the Fill Data entry number which corresponds to the module number from which the data is coming.

CRC Generation, if enabled (Normal or Invert), creates a 16 bit CCITT Cyclic Redundancy Code and inserts this value into the buffer, based on the location of the frame synchronization pattern. If the frame synchronization pattern was placed at the beginning of the buffer, the CRC value is placed into the last two bytes of the buffer, according to the buffer length; otherwise, the CRC value is placed at the very beginning of the buffer. The CRC Seed value allows the user to set an initial value for the CRC calculation, based on an agreement with the module or entity receiving the data. The Seed defaults to hexadecimal FFFF. The Byte Offset, relative to 0, allows CRC generation starting part way into the data buffer. The CRC value may be inverted, if necessary, by choosing the "Invert" option instead of "Normal" in the Control listbox.

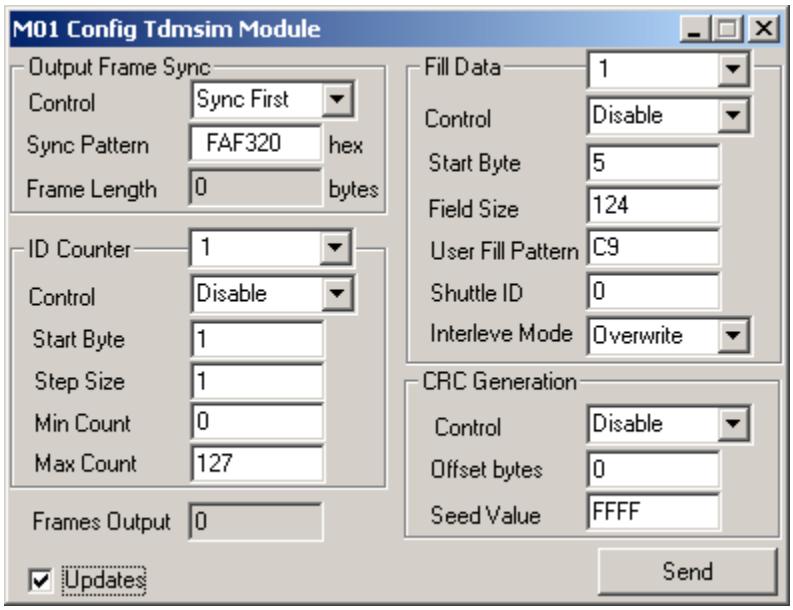
Module Window

When Tdmsim.dll is added to a desktop, the TDM Simulator dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 73-1: TDM SIMULATOR MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure TDM SIMULATOR MODULE Board Parameters sub-menu.



Module 73-2: Config TDM SIMULATOR MODULE Window

Module 73-3: TDM SIMULATOR MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
OUTPUT FRAME SYNC			
Control	User Selection	Disable, Sync First, Sync Last	Places sync at beginning (first) or end (last) of frame.
Sync Pattern	Hexadecimal	24 or 32 bit hexadecimal value; with restrictions	Configure Sync Pattern (up to 32 bits long)
Frame Length	Decimal	Display Only	Displays the frame length. The actual frame length is dictated by the length parameter in the event that triggered a frame to be generated.
ID COUNTER			
ID Counter	User Selection	1 to 30	Up to 30 ID counters can be inserted in frame – this parameter selects which ID Counter to define or display. Used as a subscript by all the following parameters for this section.
Control	User Selection	Disable Static, Increment, Decrement	Disable causes this definition (1 of 30) to be ignored. The id counter may increment or decrement. Select static to insert the same value in all frames, or to stop the id counter while data continues.

Parameter	Unit	Range or Values	Description
Start Byte	Decimal	1 to the frame length	Byte number, relative to 1, to insert this (1 of 30) counter. If the counter is defined to have a maximum value of 255 or less, only the Start Byte will be changed. If the counter has a maximum value between 256 and 65535, the counter will be inserted into Start Byte and Start Byte + 1.
Step Size	Decimal	1 to 32768	Value to increment (or decrement) this counter
Min Count	Decimal	0 to 65535	Minimum value of this counter (start value). In the case of a decrementing counter, this value would be the end value (the value after which a wrap to the maximum value occurs).
Max Count	Decimal	0 to 65535	Maximum value of counter (end value) after which a wrap to the minimum value occurs. In the case of a decrementing counter, this would be the start value.
FILL DATA			
Fill Data	User Selection	1 to 30	Up to 30 fill patterns can be inserted into frame – this parameter selects which fill pattern to configure for definition or display. The value of this item is used as a subscript for all of the following parameters in this section.
Control	User Selection	Disable, Dynamic, Byte Position, ID Counter, User Specified, Normal I/L Shuttle I/L	Disable causes this definition (1 to 30) to be ignored. Dynamic provides changing data to the user, see text for exact description. Byte position sets each byte in the fill field to a value of the actual byte position in the buffer, relative to zero. ID Counter causes the least significant 8 bits of the correspondingly defined counter (in the ID Counters section) to be loaded into the each and every byte in the field. User Specified loads the hexadecimal value specified into each and every byte in the field. Normal Interleave loads or exclusive-ORs bytes sent to this module by the corresponding module number into this field. Exclusive-Oring assumes that the field has previously been set to some initial value with which the XOR operation occurs. Shuttle Interleave loads or exclusive-Ors bytes sent to this module by the corresponding module number into this field, prepended with the PB1 time and Shuttle ID. Exclusive-Oring assumes that the field has previously been set to some initial value with which the XOR operation occurs. The PB1 is retrieved from the PB4 time associated with the buffer of the sending module. The Shuttle ID is taken from the gui.
Start Byte	Decimal	1 to size of buffer, 1 = most significant byte	Select byte in frame to insert start of fill pattern
Field Size	Decimal	1 to size of buffer	Size of fill pattern in bytes

Parameter	Unit	Range or Values	Description
User Fill Pattern	Hexadecimal	00 to FF	Defines Fill Pattern, for each byte in the field to be used when Control parameter is set to User Specified. Also used to initialize the buffer for Normal I/L, or Shuttle I/L exclusive-OR operation.
Shuttle ID	Decimal	0 to 31	Shuttle ID value; when loaded into the interleave buffer, this value is shifted left three bits. The least significant bits are set to binary 100.
Interleave Mode	User Selection	Overwrite, XOR	Overwrite or XOR: Defines method of interleaving. Overwrite inserts this interleave stream from the corresponding module number into the fill field. XOR causes the data to be XORed with the content of the buffer. The user needs to initialize the fill field with the necessary pattern to use in the XOR operation; use a lowered number fill operation such as User Specifiable to initialize this field.
CRC GENERATION			
Control	User Selection	Disable, Normal, Invert	Controls the generation of a 16 bit CCITT codeword to be inserted into the buffer. See text for full description. The result is stored in the buffer if set to Normal, Invert causes the code word to be inverted prior to storage in the buffer.
Offset Bytes	Decimal	0 to size of buffer, 0 is the most significant byte	Byte offset to begin CRC calculation.
Seed value	Hexadecimal	0000 to FFFF, default = FFFF	Initial value used to Seed the CRC generation
OTHER SETUP			
Frames Output	Decimal	Display Only	Number of frames output by this module

Extended Options Window Definition

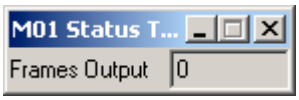
There is no extended options window for the TDM SIMULATOR MODULE.

Configuration Notes

There are no configuration notes for the TDM SIMULATOR MODULE.

Status Window Definition

When you click the Status button on the TDM SIMULATOR MODULE window, a Status TDM SIMULATOR window appears, as shown below.



Module 73-4: Status TDM SIMULATOR MODULE Window

Module 73-5: TDM SIMULATOR MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Frames Output	Decimal	Display Only	Displays number of frames that have been transmitted from this module

Module 74 TIME DELAY TEST MODULE

MODULE NAME: TimeDelayTest.dll

Key Functions

- Performs full loop delay/jitter testing
- Measure Bit Error performance
- Measures effective bit rates
- Creates performance data base

Required Hardware

- This module requires no special hardware to be installed in the Avtec PTP for Windows in order to run.

Add Module Guidelines

This module may be loaded any number of times.

Required Input Source, Connections, and Events

INPUT SOURCE: Any module?

CONNECTIONS: When data logging is enabled, statistics are output on port 2 and that port must be connected to a File Recorder Module. Statistics are logged for each frame input.

EVENTS: None.

Module Functional Description

TBS?

Input Data Format

TBS?

Output Data Format

Statistics can be generated as a comma delimited data base file.

Output port2 should be sent to FileRecorder.dll.

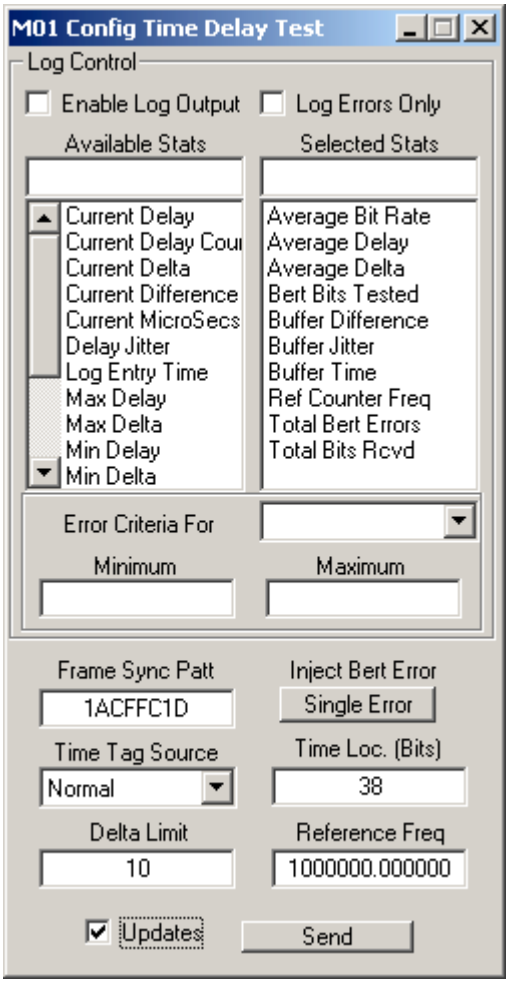
Module Window

When TimeDelayTest.dll is added to a desktop, the Time Delay Test dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 74-1: Configure TIME DELAY TEST MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure TIME DELAY TEST MODULE Board Parameters sub-menu.



Module 74-2: Config TIME DELAY TEST MODULE Window

Module 74-3: TIME DELAY TEST MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
LOG CONTROL			
Enable Log Output	Selectable Setup State	Enable Disable	As described in the functional description, this module can be used to log information to file. Log information is output on port 2. This field enables the log output function. When enabled, all statistics chosen in the Selected Stats window are logged to file. This is a check box setup: insert a check to enable and remove the check to disable.
Log Errors Only	Selectable Setup State	Enable Disable	Not currently implemented.
Available Stats	Selectable Setup Parameters	See description	The available stats list is all the statistics that the module can be set up to maintain during a data flow. To select a statistic, click on the statistic name and it moves over to the Selected Stats window. Once selected, it no longer shows up in the Available Stats window. The available statistics and a brief description of each follows: 1) Average Bit Rate 2) Average Delay 3) Average Delta 4) Bert Bits Tested 5) Buffer Difference 6) Buffer Jitter 7) Buffer Time 8) Current Delay 9) Current Delay Count 10) Current Delta 11) Current Difference 12) Current Microseconds 13) Delay Jitter 14) Log Entry Time 15) Max Delay 16) Min Delay 17) Prev MicroSecs 18) Ref Counter Freq 19) Ref Counter Offset 20) Ref Scaling Factor 21) Remote Delay Count 22) Total BERT Errors 23) Total BERT Slips

Parameter	Unit	Range or Values	Description
Selected Stats			Once an available statistic is chosen, it shows up in this window and the module is configured to maintain the count associated with that statistic. To remove a selected statistic, highlight the statistic name and it automatically moves out of the selected window and back into the Available Stats window.
Error Criteria For			Not currently implemented.
Minimum			Not currently implemented.
Maximum			Not currently implemented.
OTHER SETUP			
Frame Sync Patt	hex		Sync Pattern to be inserted in transmitted TimeDelay frames. Filter for rcvd TimeDelay frames. Setting this value to all 0's will cause all rcvd data to be treated as unformatted continuous 2047 pattern as transmitted from the BertModule. This allows TimeDelay test to function simply as a Bit Error Performance tester.
Inject Bert Error	Issues command	Issues once and stops	Injects one bit error in the transmitted data.
Time Tag Source	Selectable Setup State	See description	Determines module's mode of operation. This is the key setup parameter for module operations-- it determines what time tag format is being used and can set the module up to compare two time tag values that are located in known positions. The most commonly used modes of operation are Normal and Buffer Time, both of which use the standard modules and time tag formats. Refer to the list below for a description of each mode of operation. The options for time tag source are as follows: 1) Normal: 2) Buffer Time: 3) TXE Time Tag: 4) Compare TXE: 5) Soft Time Tag: 6) Compare Soft:
Time Loc	Bits		Points to the relative start of the time field to extract, if the input data is a telemetry stream containing TXE ?????????? Time tagging.
Reference Frequency			TBS

Extended Options Window Definition

There is no extended options window for the TIME DELAY TEST MODULE.

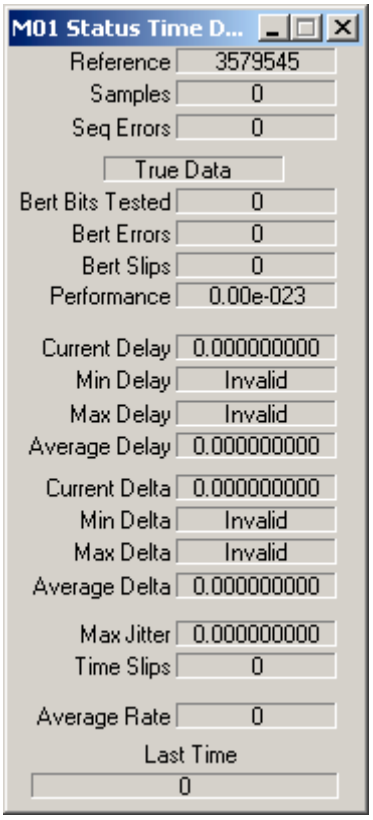
Configuration Notes

When you click the Command button on the TIME DELAY TEST MODULE window, a Command TIME DELAY TEST window appears, as shown below.

Module 74-4: Command TIME DELAY TEST MODULE Window

Status Window Definition

When you click the Status button on the TIME DELAY TEST MODULE window, a Status TIME DELAY TEST window appears, as shown below.



Module 74-5: Status TIME DELAY TEST MODULE Window

Module 74-6: TIME DELAY TEST MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Reference			TBS
Samples	Frames	0 to 65535	Number of TimeDelay frames tested.

Parameter	Unit	Range or Values	Description
Seq Errors	Frames	0 to 65535	Number of TimeDelay frames with sequence errors.
Bert Bits Tested	Bits	64 bits	Total number of 2047 pattern bits rcvd.
Bert Errors	Bits	64 bits	Number of 2047 pattern bits that were in error.
Bert Slips	Frames	64 bits	Number of times pattern lock was lost.
Perform-ance		float	Releative performance index. BertBitsTested / BertErrors
Current Delay	Seconds	Double float	In NORMAL mode the time difference from the transmitted frame to the actual rcvd time. In BUFFERTIME mode the difference between local PTPTimeStamps of consecutive frames
Min Delay	Seconds	Double float	Latched minimum difference of CurrentDelay - PreviousDelay.
Max Delay	Seconds	Double float	Latched maximum difference of CurrentDelay - PreviousDelay.
Average Delay	Seconds	Double float	Total CurrentDelay of all Samples / number of Samples
Current Delta	Seconds	Double float	Rate of change -- CurrentDelay – PreviousDelay.
Min Delta	Seconds	Double float	Latched minimum difference of CurrentDelta - PreviousDelta.
Max Delta	Seconds	Double float	Latched maximum difference of CurrentDelta - PreviousDelta.
Average Delta	Seconds	Double float	Total CurrentDelta of all Samples / number of Samples
Max Jitter	Seconds	Double float	Max delay minus Min delay
Time slips	Frames	64 bits	Number of frames where time did not increment or regressed.
Average Rate	Bits per second	64 bits	In all modes this will measure the effective bit rate of rcvd data.
Last Time	Varies		In NORMAL mode this will be an absolute count of elapsed system time. Otherwise it will be a decoded PB4 format of the last rcvd time stamp.

Module 75 TLM-3-29 STANDARD FORMATTED DATA UNIT (SFDU) MODULE

MODULE NAME: Sfdu3_29Formatter.dll

Key Functions

- Formats input data. Appends TLM-3-29 SFDU Header to each input buffer and then outputs the SFDU
- Provides a Decoder Mode where the module decodes each input SFDU and displays the header field values. Does not provide formatting in this mode.
- Automatically updates SFDU fields based on input buffer status, e.g., synchronization status, RS status, data lengths, data rates, sequence number
- Optionally provides a time out message that is output when data input has stopped for a programmed length of time
- Optionally provides periodic messages output that are continuously output at a programmable rate when data is not input to module.

Required Hardware

- This module does not require any special hardware installed in the Avtec PTP for Windows system.

Add Module Guidelines

Do not exceed the 32 modules per desktop limit. Add one module for each virtual stream required unless virtual stream ID is based on VCID.

Required Input Source, Connections, and Events

INPUT SOURCE: This module receives buffered input from another module.

CONNECTIONS: The module must be connected to both an input and output source/destination module. This module outputs to port one.

EVENTS: No event inputs are required.

Module Functional Description

The SFDU3_29FORMATTER MODULE provides two modes of operation: Formatter and Decoder. The Formatter mode is the normal mode of operation that provides SFDU formatting. The Decoder mode provides status information only and does not perform any processing on the input data.

In the Formatter Mode, input data is formatted into SFDUs and output from the module. For each buffer input, the module prepends the TLM-3-29 header and then outputs the SFDU (header plus buffer) to port one. The SFDU format is defined by the "TLM–3–29 Telemetry Standard Formatted Data Unit (SFDU) Interface, 820-013 Deep Space Mission System

External Interface Specification, JPL D-16765" Document, which was written by JPL. Refer to that document for details of the SFDU format and individual bit field definitions. The SFDU header length is 100 bytes.

The SFDU3_29FORMATTER MODULE Format Mode is designed to work with input provided from another module so that many of the header fields can be automatically filled to accurately reflect the status of the frame contained in that SFDU. For example, data that is processed by the Serial Input Module (Monarch E), and then input to the SFDU3_29FORMATTER MODULE includes status about frame synchronization, Reed-Solomon decoding, and the frame synchronization strategy that was used to obtain that frame. Although this automatic field creation is available, for all SFDU header fields, the operator has the option of selecting manual configuration, which puts an operator-defined fixed value at that header bit location.

In Decoder Mode, the module expects input data to be in TLM-3-29 SFDU format. This is a status only mode that can be used for data verification or debugging, and the module displays the input data header field values to the Status Window.

The SFDU3_29FORMATTER MODULE provides two types of status messages that can optionally be enabled: Time-Out Status Message and Periodic Time Out Messages. Each type of message is independently enabled, which means either or both can be enabled. Both types of messages require the operator to configure a time out value in seconds.

1) Time-out Status Message: A single SFDU is generated to indicate that frame synchronizer is out-of-lock. A Time-out Status Message is generated when the SFDU module stops receiving data. Therefore, this message indicates loss of lock in frame synchronous mode: it indicates that user was receiving data, and then frame sync went out of lock. If status is enabled when the system is in throughput mode, this message indicates that clock and data was present, and then the clock stopped. When enabled, the Time-out Status SFDU does not apply until data input is started. After data input, if data stops and the time-out period (user programmed) passes without more data input, one status message is generated. Another status message will not be generated unless data starts again, and then stops for the programmed time out interval. A Time-out Status Message has the following characteristics:

   - Virtual stream value is 127,
   - It has it's own sequence counter, ie, the sequence counter starts at 1 and increments for each SFDU with virtual stream ID 127.
   - Telemetry length is zero, and all other length attributes are adjusted to indicate no telemetry.
   - Frame sync status flags indicate frame sync out-of-lock.

2) Periodic Time-out Message: The Periodic Time-out SFDU message behaves like a CAB and insures that data is always present on that channel. This type message does not wait for data to start. Instead as soon as it is enabled and the programmed time-out value is

reached, a status message is generated. Status messages continue to be generated at the time-out interval unless data is received. Any time the programmed time-out interval is exceeded without data input, another Periodic Time-out Message is generated. This means that if the periodic time out value is programmed is less than the data rate, then the Periodic time out message is intermingled with telemetry data flow. A Periodic Time-out Message has the following characteristics:

- Virtual stream value is 126,
- It has it's own sequence counter, ie, the sequence counter starts at 1 and increments for each SFDU with virtual stream ID 126.
- Telemetry length is zero, and all other length attributes are adjusted to indicate no telemetry.
- Frame sync status flags indicate frame sync out of lock.

The following table defines the SFDU header contents. The table accurately reflects the order of each SFDU section and the number of words contained in each. The total header length is 100 bytes (50 words). The total SFDU length is header length plus the input buffer size. If the input buffer is an odd length, it is padded out to 16-bit word boundary.

Module 75-1: TLM 3-29 SFDU Layout

TLM 3-29 Telemetry SFDU Layout		
Word (16 bit/word)	**Field Name/Description**	**Length (16 bit word)**
1 to 10	SFDU Label	10
11 to 12	Header Aggregation Label	2
13 to 16	Primary Header	4
17 to 48	Secondary Header	32
49 to N	Telemetry Data	N-48

Input Data Format

In Formatter Mode, input to this module is Avtec PTP for Windows data buffers that are to be put in SFDU format, and then output. Typically in this mode, input data is frame synchronized and Reed-Solomon Decoded. The input has often been sorted by VCID prior to Input to the module. Sorting by VCID and routing to multiple SFDU Modules provides control of Virtual Stream content and allows unique assignment of Virtual Stream Identification.

If the input data has been processed by another module prior to input to this module, then the input Avtec PTP for Windows data buffer includes status information. The associated status information is used to fill in the SFDU header. In a typical scenario, data is processed by the Monarch E (Serial Input Module) and CCSDS Virtual Channel Processor Version 2 prior to input to the TLM3-29 SFDU Formatter Module. For this scenario, extensive frame synchronization and Reed-Solomon mode (decoder enabled/disabled) and status information is available to be used in the SFDU header.

Although it is typical that input is frame synchronized, the module formats each input buffer regardless of that buffers content. As a result, the input does not have to be frame synchronized and can be raw bit data.

When the module is set to Decoder Mode, the input is data that is already in SFDU format.

Output Data Format

In Formatter Mode, output data is in SFDU format. SFDU format is an SFDU header appended to each Avtec PTP for Windows buffer input to the module.

In Decoder Mode, the output is throughput of the input. It is expected that the input is data in SFDU format, and therefore the output is also in SFDU format.

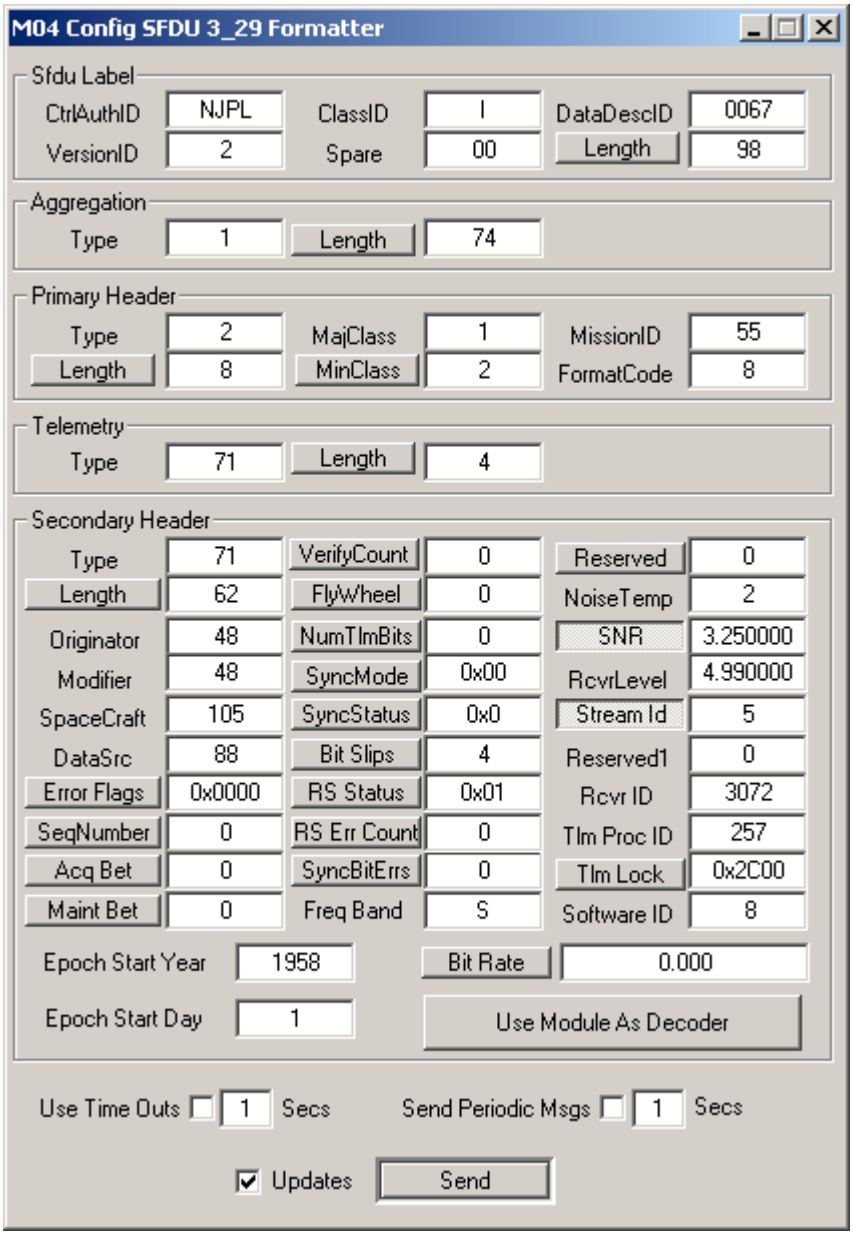
Module Window

When *Sfdu3_29Formatter.dll* is added to a desktop, the SFDU 3-29 Formatter dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 75-2: SFDU 3-29 FORMATTER MODULE Window

Configure Window Definition

When the Configure button is selected, a Config window appears, as shown below. The table that follows defines the fields in the Configure SFDU3_29FORMATTER MODULE.



Module 75-3: Config SFDU 3-29 FORMATTER MODULE Window

Module 75-4: SFDU 3-29 FORMATTER MODULE Configure Window Parameters

CONFIGURE WINDOW

The Configuration Window provides a series of data field entries that the operator configures. The available fields directly map to the TLM–3–29 Telemetry Standard Formatted Data Unit (SFDU) Interface Document and the fields are grouped according to the SFDU header section to which they are mapped (eg SFDU Label, Primary Header, Secondary Header). Together the header sections comprise the complete SFDU header, but the Configuration Window does not list each section in word order. Specifically, the Telemetry Data Field is the last section of the header prior to the telemetry data (telemetry data is the Avtec PTP for Windows input buffer).

Every data field entry that is prefaced by a button can be configured to either calculate the data field automatically (automatic) or to fill the field with a fixed value (manual). When in manual configuration, each output SFDU header field is filled with the user-defined fixed value that is entered after the data field name button. When in automatic mode, that field is filled in using the status information that accompanies the input buffer (which requires that the input buffer was processed by other module prior to input to this module) or it is filled in based on information calculated or maintained by the SFDU 3_29 Module. For example, when set to automatic, frame synchronization and Reed-Solomon mode and status are filled in based on input buffer status.

To configure a field to be manual, which inserts a fixed value at that location into every output SFDU, the button must be enabled (depressed).

The Secondary Header Sequence Field is different from all other fields in this window. It is a three state field, and the button must the set to the desired method for calculating SFDU sequence numbers.

For purpose of these definitions, a word is a 16-bit value. Bit definitions are left to right with the left-most bit being bit 1.

For each SFDU configuration field, the description provides the Word location of that field (relative to the start of that SFDU section) and the bits of the word to which that field is mapped. If the description simply provides the word location then that field maps to all 16 bits of that word. Although the TLM-3-29 has expected/standard configurations for certain fields, any time the user makes a fixed-manual setup, the field limits are defined by the number of bits to which that field maps.

Several options on this window control the module mode of operation and top-level functionality. As a first step to setup, the operator should insure that these fields are correctly set. The mode/functionality fields are provide first in the following detailed descriptions.

MODE/MODULE FUNCTIONAL DEFINITIONS

Parameter	Unit	Range or Values	Description
Use Module As Decoder	configuration parameter	enable disable	When enabled, the module does NOT PROVIDE formatting, and the module expects SFDU input. In decoder mode, it decodes the data fields of each SFDU header input to the module and displays the field values to the status window.
Epoch Start Year	year	users-defined	manual entry TLM-3-29 format expects the entry to be 1958. This is the absolute start year. Time stamp (for SFDU ERT) is based on this start year. Time stamp is relative to this start date.
Epoch Start Day	day	users-defined	manual entry Default value is 1. This is the absolute start day. The field is the Julian Day of Year. This field is used in conjunction with Epoch Star Year to define the ERT time stamp day 0. For the provided example, January 1, 1958 is day zero, which means that ERT day 1 is January 2, 1958.

Bit Rate	bps	auto is based on Serial In. Module status	manual or automatic entry When set to automatic, the value entered in this field is the bit rate that is input with the Avtec PTP for Windows buffer. Typically, this is receive bit rate calculated by the Monarch E board.
Use Time Outs	configuration/ seconds	enabled disabled	This option can optionally be enabled. When a check is entered, then this function is enabled. Time out value is entered in seconds. The time out value determines the amount of time that must pass without data input to the module before a time out message is output from the module.
Send Periodic Messages	configuration/ seconds	enabled disabled	This option can optionally be enabled. When a check is entered, then this function is enabled. Output rate is entered in seconds. This is the rate at which Periodic Status Messages are output when no data is input to the module.
SFDU HEADER DEFINITIONS			
Parameter	**Unit**	**Range or Values**	**Description**
SFDU LABEL-- 10 WORDS			
CtrlAuthID	ascii	user-defined text entry	manual entry Control authority identifier, Typical value NJPL Word 1 and 2 of SFDU Label
VersionID	ascii	user-defined text entry	manual entry SFDU label version ID; "2" indicates that the length attribute field in words 7 to 10 is formatted as a binary unsigned integer Bits 1 to 8/Word 3 of SFDU Label
ClassID	ascii	user-defined text entry	manual entry SFDU class ID; Typical value "I" Bits 9 to 16/Word 3 of SFDU Label
Spare	hex	user-defined	manual entry Reserved Word 4 of SFDU Label
DataDescID	ascii	user-defined text entry	manual entry Data description identifier; Typical value '0067' ascii Word 5 and 6 of SFDU Label
Length	decimal	real-time calc. or user-defined entry	manual user entry or automatic calculation Length attribute of the telemetry SFDU. Indicates the length, in octets, of the value field of the telemetry SFDU, words 11 through N. When automatic, this value accurately reflects the SFDU length, which is defined by the input buffer length.. Word 7 to 10 of SFDU Label
HEADER AGGREGATION-- 2 WORDS			

Type	decimal	user-defined	manual entry Type Attribute-- Typical Value= 1 Word 1 of Header Aggregation
Length	decimal	automatic or user-defined entry	manual user entry or automatic calculation Length Attribute-- Indicates the length in octets of the value field of the header aggregation (words 13 through 48 of the SFDU). Word 2 of Header Aggregation
PRIMARY HEADER-- 4 WORDS			
Type	decimal	user-defined	manual entry Type Attribute-- Typical Fixed value is 2 Word 1 of Primary Header
Length	decimal	automatic, or user-defined	manual user entry or automatic calculation Length Attribute-- Fixed value 4 indicates the length in octets of the value field of the primary header Word 2 of Primary Header
MajClass	decimal	user-defined	manual entry Major Data Class Value = 1 (indicates this SFDU contains spacecraft telemetry.) Bits 1 to 8/Word 3 of Primary Header
MinClass	decimal	user-defined or automatic	manual user entry or automatic calculation Minor Data Class Field Values = 7, 8, or 9 Indicates the configuration (enabled/disabled) of the frame synchronization and RS decoding when the data in this SFDU was acquired: 7= FS and RS disabled; 8= FS enabled, RS disabled; 9 = FS and RS enabled. When automatic, then this field is filled in based on the input data processing (of another module) prior to this module. Bits 9 to 16/Word 3 of Primary Header
MissionID	decimal	user-defined	manual entry Mission Identifier Range Identifies the mission whose telemetry is contained in this SFDU (this is NOT the SCID). Values = 0 to 255 Bits 1 to 8/Word 4 of Primary Header
FormatCode	decimal	user-defined	manual entry Format Code Typical Value = 0, which indicates this is a DSS-formatted SFDU Bits 9 to 16/Word 4 of Primary Header
TELEMETRY DATA--2 WORDS PLUS DATA			

Type	decimal	user-defined	manual entry Type attribute of the telemetry data: 10 indicates binary data Word 1 of Telemetry Data
Length	decimal	automatic, or user-defined	manual or automatic entry Length attribute Word 2 of Telemetry Data
SECONDARY HEADER--32 WORDS			
Type	decimal	user-defined	manual entry Type Attribute Word 1 of Secondary Header
Length	decimal	automatic, or user-defined	manual or automatic entry Length Attribute: This is the length in octets of the value field of the secondary header. Typical value is fixed 0x3C = 60 Word 2 of Secondary Header
Originator	decimal	user-defined	manual entry Originator ID: Typical value = 0x30 = 48 = DSN Bits 1 to 8/Word 3 of Secondary Header
Modifier	decimal	user-defined	manual entry Last modifier ID: Typical value = 0x30 = 48 = DSN Bits 9 to 16/Word 3 of Secondary Header
SpaceCraft	decimal	user-defined	manual entry DSMS assigned Spacecraft ID (NOT the same as CCSDS SCID) Value = 0 to 255 Bits 1 to 8/Word 4 of Secondary Header
DataSrc	decimal	user-defined	manual entry Data Source Identifier Value = 0 to 255 Bits 9 to 16/Word 4 of Secondary Header

ErrorFlags	hex	automatic, or user-defined	manual user entry or automatic calculation Word 5 of Secondary Header When automatic, bit 14 is set to 1; bits 6 and 7 are set to one; and all other bits reflect the status of the input buffer. Bit 1 to 4 = Reserved Bit 5 ERT Reference point 0= ERT reference trailing edge of the last received telemetry bit in the SFDU 1= leading edge Bit 6 = ERT extended resolution field status 0= ERT extended resolution field is reserved and its value should be ignored 1= ERT extended resolution field is valid Bit 7 = If bit 6 is 0, then bit 7 is set to 0. Otherwise, this bit Indicates the units of the ERT extended resolution field 0= A value of '0' indicates that the ERT extended resolution field contains the number of tenths-of-microseconds of the millisecond. 1= A value of '1' indicates that the ERT extended resolution field contains the number of microseconds of the millisecond. Bit 8 = ERT status: 0= valid ERT 1= ERT invalid Bits 9 to 13 = reserved Bits 14 = 1= SNR value in words 23 and 24 is in bit domain
SeqNumber	decimal	automatic, or user-defined	manual user entry or automatic calculation When in automatic, the record sequence number: starts with 1 and increments for each SFDU based on stream ID. Separate sequences are maintained for each stream ID. If the stream ID is a fixed value (see stream ID definition), then the sequence number increments by one for each SFDU output. If the stream ID is based on input frame VCID, then the sequence number is separated for each VCID. Global Virtual Stream Identifier Word 10 and 11 of Secondary Header
Acq Bet	decimal	automatic, or user-defined	manual user entry or automatic calculation Acquisition bit error tolerance; number of allowed bit errors in the sync during search and check. When automatic, the data must have been processed by the Serial Input Module (Monarch E), and then this value reflects the search/check bit error tolerance. Bits 1 to 8/Word 12 of Secondary Header

Maint Bet	decimal	automatic, or user-defined	manual user entry or automatic calculation Maintenance bit error tolerance; number of allowed bit errors in the sync during lock and flywheel When automatic, the data must have been processed by the Serial Input Module (Monarch E) prior to input, and then this value reflects the lock/flywheel bit error tolerance. Bits 9 to 16/Word 12 of Secondary Header
Verify Count	decimal	automatic, or user-defined	manual user entry or automatic calculation Verify count; number of within-tolerance frames required during check mode to transition to lock mode. When automatic, the input data must be processed by the Serial Input Module (Monarch E), and then this value reflects the Serial Input Module check frame configuration. Bits 1 to 8/Word 13 of Secondary Header
Flywheel	decimal	automatic, or user-defined	manual user entry or automatic calculation Flywheel count; number of out-of-tolerance frames required during flywheel mode to transition to search mode When automatic, the input data must be processed by the Serial Input Module (Monarch E), and then this value reflects the Serial Input Module flywheel frame configuration. Bits 9 to 16/Word 13 of Secondary Header
NumTlmBits	decimal	automatic, or user-defined	manual user entry or automatic calculation Number of received telemetry bits. When automatic, it indicates the number of actual received telemetry bits that are contained in the value field of the telemetry data. If the number of received telemetry bits is less than the length of the value field of the telemetry data, unused bits appear at the end of the field and shall be ignored. Word 14 of Secondary Header
SyncMode	hex	automatic, or user-defined	manual user entry or automatic calculation Bits 1 to 8/Word 15 of Secondary Header When automatic, flags are based on input buffer status. Most commonly, the data will have been processed by the Serial Input Module, and these flags reflect that modules setup and state of processing at the time the buffer of data was generated. Bits 1 to 8-- Frame Sync Mode Flags Bit 1=Operator Forced Resync Bit 2=Reserved Bit 3=APC is enabled Bit 4 =Frame Sync in flywheel mode Bit 5=Frame Sync in lock mode Bit 6=Frame Sync in check mode Bit 7=Frame Sync in search mode Bit 8=Frame Sync in Bypass mode (1=synchronizer disabled).

Sync Status	hex	automatic, or user-defined	manual user entry or automatic calculation Bit 9/Word 15 of Secondary Header -- Data Polarity Flag 0=true sync marker detected. 1=inverted sync marker detected
Bit Slips	decimal	automatic, or user-defined	manual user entry or automatic calculation Bits 10 to 16/Word 15 of Secondary Header-- bit slip flags that are not implemented in Avtec PTP for Windows
RS Status	hex	automatic, or user-defined	manual user entry or automatic calculation Bits 1 to 8/Word 16 of Secondary Header When automatic, the data has been RS decoded by either the Serial Input Module (Monarch E hardware decode) or the CCSDS Virtual Channel Processor Module (software decode). Bits 1 to 8: RS Decoder Status 0=No RS decoding performed 1=No symbol errors detected 2=Corrected RS errors in frame 3=Uncorrectable RS errors
RS Err Count	decimal	automatic, or user-defined	manual user entry or automatic calculation Number of corrected RS symbol errors in this frame When automatic, this reflects the frame processing prior to input to SFDU module. Note that if the input data is processed by both the Serial Input Module, and then the CCSDS Virtual Channel Processor prior to input, if the Serial Input Module was set to RS decode and correct AND the VC Processor is set to RS Decode, the VC Processor overwrites Serial Input Status and all correctable error information is lost (since the Serial Input Module corrected the error). Bits 9 to 16/Word 16 of Secondary Header
SyncBitErrs	decimal	automatic, or user-defined	manual user entry or automatic calculation Attached Sync Marker Bit Error Count: When automatic, indicates the number of bit errors in the attached sync marker detected by the frame synchronizer. Bits 1 to 8/Word 17 of Secondary Header
FreqBand	ascii	user-defined text entry	manual entry Frequency Band of the master receiver. Bits 9 to 16/Word 17 of Secondary Header
Reserved	decimal	user-defined	manual user entry or automatic calculation Reserved Word 20 of Secondary Header
Noise Temp	decimal	user-defined	manual entry System noise temperature Word 21 and 22 of Secondary Header

SNR	floating point	automatic, or user-defined	manual user entry or automatic calculation Estimated signal to noise ratio in db Word 23 and 24 of Secondary Header
RcvrLevel	floating point	user-defined	manual entry Receiver signal level: master receiver Word 25 and 26 of Secondary Header
Stream ID	decimal/ user-defined setup	Stream ID, VCID, AXAFID	Three state configuration button: Stream ID, VCID, AXAF ID. Stream ID: The user provides a fixed value that is stamped into every SFDU output from module. VCID: (automatic) The value at this location reflects the VCID of the input frame. The field is the input frame VCID. AXAF ID: (automatic) The module expects the input frame to have a VCID value that is in AXAF format, which means that the VCID is located in the top three bits of the frame VCID field. The SFDU value in this field reflects the input frame (AXAF) VCID. Virtual Stream Identifier Field Bits 1 to 8/Word 27 of Secondary Header
Reserved 1	decimal	user-defined	manual entry Reserved Bits 9 to 16/Word 27 of Secondary Header
Rcvr ID	decimal	user-defined	manual entry Receiver Identifier Word 28 of Secondary Header
Tlm Proc ID	decimal	user-defined	manual entry Telemetry Processor Identifier Word 29 of Secondary Header
Tlm Lock	hex	automatic, or user-defined	manual user entry or automatic calculation Word 30 of Secondary Header Telemetry Lock Status Code; each bit indicates the lock status of selected equipment Bits 1 and 2 = Carrier demodulation Bits 3 and 4 = Reserved Bits 5 and 6 = Subcarrier demodulator Bits 7 and 8 = Symbol synchronizer Bits 9 and 10 Convolutional Decoder Bits 11 and 12 Frame Synchronizer (lock and flywheel) Bits 13 and 14 Reed-Solomon Decoder Bits 15 AND 16 Reserved 00=not in use/01=invalid code/10=in lock/11=out-of-lock

Software ID	ascii	user-defined	manual entry Telemetry Software Identifier Word 31 of Secondary Header

Extended Options Window Definition

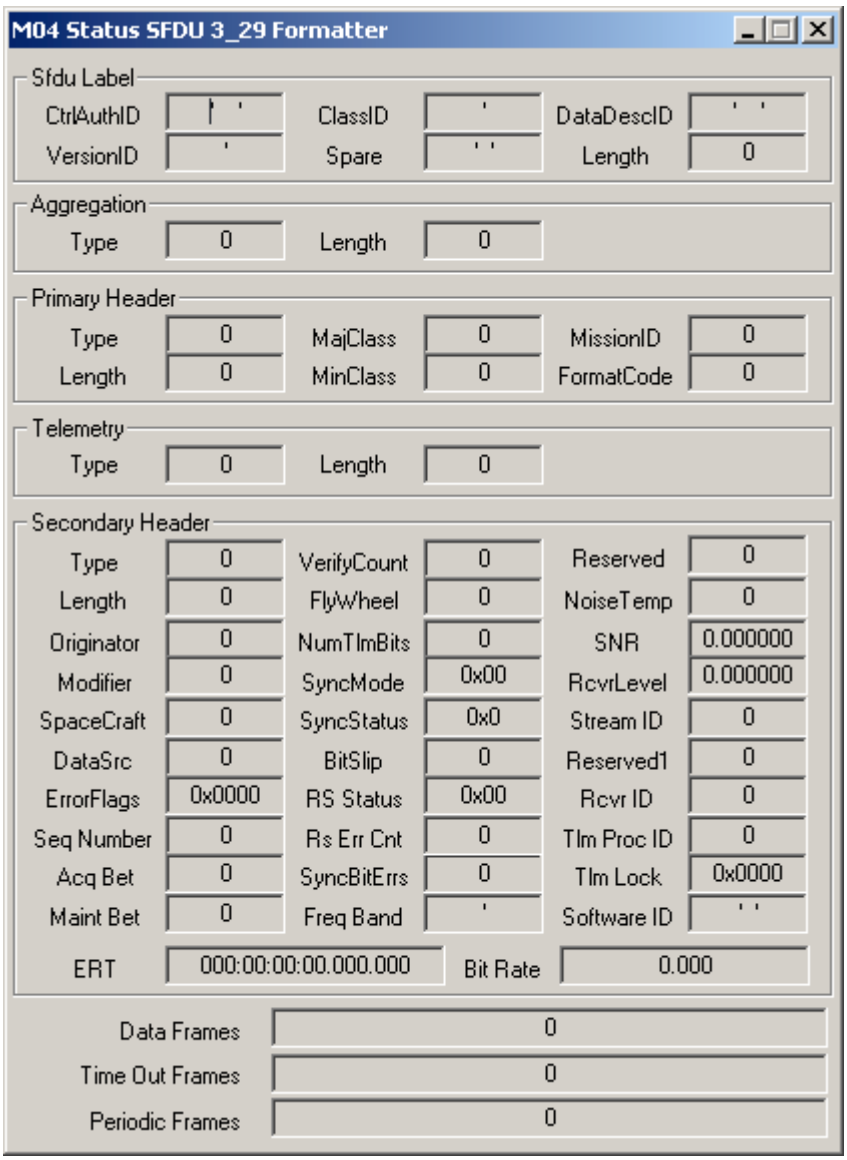
There is no extended options window for the SFDU3_29FORMATTER TRANSFER MODULE.

Configuration Notes

There are no configuration notes for the SFDU3_29FORMATTER TRANSFER MODULE.

Status Window Definition

When you click the Status button on the SFDU3_29FORMATTER MODULE window, a Status SFDU3_29FORMATTER MODULE window appears, as shown below.



Module 75-5: Status SFDU 3-29 FORMATTER MODULE Window

Module 75-6: SFDU 3-29 FORMATTER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
The status values reflect the bits read from that header location. For significance/definition of each parameter, refer to the configuration window definitions.			
SFDU LABEL			
CtrlAuthID	ascii	32 bits	Word 1 and 2 of SFDU Label
VersionID	ascii	8 bits	Bits 1 to 8/Word 3 of SFDU Label
ClassID	ascii	8 bits	Bits 9 to 16/Word 3 of SFDU Label
Spare	hex	16 bits	Word 4 of SFDU Label
DataDescID	ascii	32 bits	Word 5 and 6 of SFDU Label

Parameter	Unit	Range or Values	Description
Length	decimal	32 bits	Word 7 to 10 of SFDU Label
AGGREGATION			
Type	decimal	16 bits	Word 1 of Header Aggregation
Length	decimal	16 bits	Word 2 of Header Aggregation
PRIMARY HEADER			
Type	decimal	16 bits	Word 1 of Primary Header
Length	decimal	16 bits	Word 2 of Primary Header
MajClass	decimal	8 bits	Bits 1 to 8/Word 3 of Primary Header
MinClass	decimal	8 bits	Bits 9 to 16/Word 3 of Primary Header
MissionID	decimal	8 bits	Bits 1 to 8/Word 4 of Primary Header
FormatCode	decimal	8 bits	Bits 9 to 16/Word 4 of Primary Header
TELEMETRY			
Type	decimal	16 bits	Word 1 of Telemetry Data
Length	decimal	16 bits	Word 2 of Telemetry Data
SECONDARY HEADER			
Type	decimal	16 bits	Word 1 of Secondary Header
Length	decimal	16 bits	Word 2 of Secondary Header
Originator	decimal	8 bits	Bits 1 to 8/Word 3 of Secondary Header
Modifier	decimal	8 bits	Bits 9 to 16/Word 3 of Secondary Header
SpaceCraft	decimal	8 bits	Bits 1 to 8/Word 4 of Secondary Header
DataSrc	decimal	8 bits	Bits 9 to 16/Word 4 of Secondary Header
ErrorFlags	hex	16 bits	Word 5 of Secondary Header
SeqNumber	decimal	32 bits	Word 10 and 11 of Secondary Header
Acq Bet	decimal	8 bits	Bits 1 to 8/Word 12 of Secondary Header
Maint Bet	decimal	8 bits	Bits 9 to 16/Word 12 of Secondary Header
Verify Count	decimal	8 bits	Bits 1 to 8/Word 13 of Secondary Header
Flywheel	decimal	8 bits	Bits 9 to 16/Word 13 of Secondary Header
NumTlmBits	decimal	16 bits	Word 14 of Secondary Header
SyncMode	hex	8 bits	Bits 1 to 8/Word 15 of Secondary Header
Sync Status	hex	1 bit	Bit 9/Word 15 of Secondary Header
Bit Slips	decimal	7 bits	Bits 10 to 16/Word 15 of Secondary Header
RS Status	hex	8 bits	Bits 1 to 8/Word 16 of Secondary Header
RS Err Count	decimal	8 bits	Bits 9 to 16/Word 16 of Secondary Header
SyncBitErrs	decimal	8 bits	Bits 1 to 8/Word 17 of Secondary Header
FreqBand	ascii	8 bits	Bits 9 to 16/Word 17 of Secondary Header
Reserved	decimal	16 bits	Word 20 of Secondary Header
Noise Temp	decimal	32 bits	Word 21 and 22 of Secondary Header

Parameter	Unit	Range or Values	Description
SNR	floating point	32 bits	Word 23 and 24 of Secondary Header
RcvrLevel	floating point	32 bits	Word 25 and 26 of Secondary Header
Stream ID	decimal	8 bits	Bits 1 to 8/Word 27 of Secondary Header
Reserved 1	decimal	8 bits	Bits 9 to 16/Word 27 of Secondary Header
Rcvr ID	decimal	16 bits	Word 28 of Secondary Header
Tlm Proc ID	decimal	16 bits	Word 29 of Secondary Header
Tlm Lock	decimal	16 bits	Word 30 of Secondary Header
Software ID	decimal	16 bits	Word 31 of Secondary Header
ERT		64 bits	Words 6 through 9 of Secondary Header
Bit Rate	bps	32 bits	Word 18 and 19 of Secondary Header
Data Frames	frames	incrementing counter	Total number of SFDU formatted frames output from module
Time Out Frames	frames	incrementing counter	Total number of Time Out Frames output from module
Periodic Frames	frames	incrementing counter	Total number of Periodic Frames output from module

Module 76 VEDA BIT SYNCHRONIZER MODULE

MODULE NAME: VedaBitSync.dll

Key Functions

- Control the Veda Bit Sync card.
- Displays status received from the bit sync card.

Required Hardware

- Veda Bit Sync card.
- Veda Decoder Card (if Viterbi capability is to be used)
- Appropriate cables for the card(s) to be used.

Add Module Guidelines

One module per card (or set of cards)

Required Input Source, Connections, and Events

INPUT SOURCE: All data enters the card externally from cables plugged into the back of the card.

CONNECTIONS: No module connections are made with this module.

EVENTS: No events are generated or expected.

Module Functional Description

The Veda Bit Synchronizer Module (VedaBitSync) supports both the Veda Bit Synchronizer board and the Veda Viterbi Decoder board. This module has no data inputs or outputs and cannot be "connected" to any other modules.

Input Data Format

Baseband data in user selectable format.

Output Data Format

TTL or RS422 with clock.

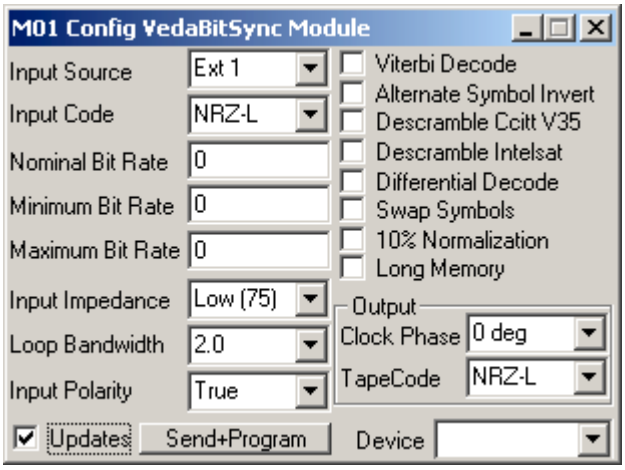
Module Window

When VedaBitSync.dll is added to a desktop, the Veda Bit Synchronizer dialog appears. Left mouse click the Configure button to view current setup or change configuration parameters. Left mouse click Status to view current module state and data processing statistics.

Module 76-1: VEDA BIT SYNCHRONIZER MODULE Window

Configure Window Definition

When you click the Configure button, a Config window appears, as shown below. The table that follows defines the fields in the Configure VEDA BIT SYNCHRONIZER MODULE Board Parameters sub-menu.



Module 76-2: Config VEDA BIT SYNCHRONIZER MODULE Window

Module 76-3: VEDA BIT SYNCHRONIZER MODULE Configure Window Parameters

Parameter	Unit	Range or Values	Description
Input Source		Internal, Ext 1, Ext 2, Ext 3, Ext 4	Specifies input. Internal or 4 External inputs are available.

Parameter	Unit	Range or Values	Description
Input Code		NRZ-L, NRZ-M, NRZ-S; BIP-L, BIP-M, BIP-S; DMM, DMS; RZ, RNRZ9-F, RNRZ11-F, RNRZ15-F, RNRZ9-R, RNRZ11-R, RNRZ15-R	Choice of input code
Nominal Bit Rate			Specifies expected bit rate
Minimum Bit Rate			Calculated using Nominal bit rate and bandwidth. May be adjusted after the Nominal rate is set.
Maximum Bit Rate			Calculated using Nominal bit rate and bandwidth. May be adjusted after the Nominal rate is set.
Input Impedance	Selectable Setup	Low (75 ohms) or Hi (10K ohms)	Impedance selection
Loop Bandwidth	Selectable Setup	0.1, 0.5, 1.0, 2.0	Choice of Loop bandwidth
Input Polarity	Selectable Setup	True Inverted	Chooses input polarity,
Viterbi Decode	Selectable Setup	Enable Disable	Enables/Disables Viterbi Decode, requires Convolutional Decoder board
Alternate Symbol Invert	Selectable Setup	Enable Disable	Enables/Disables Alternate Symbol Inversion
Descramble Ccitt V35	Selectable Setup	Enable Disable	Enables/Disables CCITT Descrambling. Only one of CCITT or Intelsat descrambling may be selected, if both are selected, both will be disabled.
Descramble Intelsat	Selectable Setup	Enable Disable	Enables/Disables Intelsat Descrambling. Only one of CCITT or Intelsat descrambling may be selected, if both are selected, both will be disabled.
Differential Decode	Selectable Setup	Enable Disable	Enables/Disables Differential Decoding
Swap Symbols	Selectable Setup	Enable Disable	Enables/Disables A and B Symbol Swap
10% Normal-ization	Selectable Setup	Enable Disable	Enable / Disable Checkbox.
Long memory	Selectable Setup	Enable Disable	See Veda Bit Sync manual for a complete hardware description.
Device			
OUTPUT			

Parameter	Unit	Range or Values	Description
Clock Phase	degree	0, 90, 180, 270	Choice of 0, 90, 180, 270 degree clock phase in relation to data. 0 causes the clock and data to be edge aligned, 180 causes the clock to be halfway out of phase with the data.
TapeCode	Selectable Setup	NRZ-L, NRZ-M, NRZ-S; BIP-L, BIP-M, BIP-S; DMM, DMS; RZ, RNRZ9-F, RNRZ11-F, RNRZ15-F, RNRZ9-R, RNRZ11-R, RNRZ15-R	Tape Output code selection

Extended Options Window Definition

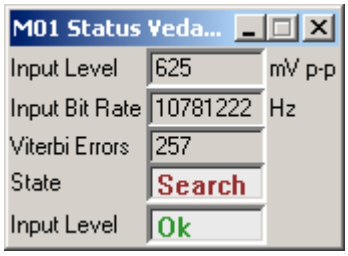
There is no extended options window for the VEDA BIT SYNCHRONIZER MODULE.

Configuration Notes

There are no configuration notes for the VEDA BIT SYNCHRONIZER MODULE.

Status Window Definition

When you click the Status button on the VEDA BIT SYNCHRONIZER MODULE window, a Status VEDA BIT SYNCHRONIZER window appears, as shown below.



Module 76-4: Status VEDA BIT SYNCHRONIZER MODULE Window

Module 76-5: VEDA BIT SYNCHRONIZER MODULE Status Window Parameters

Parameter	Unit	Range or Values	Description
Input Level	Millivolts peak-to-peak		Displays input line level as measured by the board in millivolts peak-to-peak
Input Bit Rate			Displays actual input bit rate as measured by the board

Parameter	Unit	Range or Values	Description
Viterbi Errors			Displays Viterbi errors as reported by the board (not fully implemented)
State			Displays lock state as reported by the board
Input Level			Displays input level (ok or low) as reported by the board

Appendix A:

AVTEC PTP FOR WINDOWS SERVER COMMANDS

ADD LICENSE

This command is used to register the Server software. When the software is not registered, the Server will only run in a demo mode for 20 minutes. All commands are available in this mode, however, after 20 minutes, the Avtec PTP for Windows desktop contents are cleared and a blank desktop is loaded. To register the Server software, obtain a key from Avtec Systems, Inc., run the above command from the Server command line, and input the key when asked.

ADDAPPLICATIONBUFFERS [x]

Increases the application buffer count to *x* values.

ADDEVENT [source] [destination] 0 0

This command allows an event connection to be made between two modules.

source - event source module number.
destination - event destination module number.
0 - future options, currently must be set to 0 (zero).

ADDMODULE [path\ filename]

This command opens the *Add Module* GUI, allowing the user to select a module to load. Typing the module name with the command loads that particular module. The module name must consist of the complete path (location) of the module and the module's filename. For example, to load the Serial Input Module, the command would be:

ADDMODULE C:\PTP_NT\MODULES\M0201_AvtecSerialInput.dll

The above command could be used within a Script File to load the Serial Input Module without having to go through the *Add Module* GUI box.

ADDOUTPUT [source] [destination] [port#] [input]

This command allows a data flow connection to be made between two modules.

source - data stream source module number.
destination - data stream destination module number.
port# - port number of source module to send data from. Valid values are 1 - 32.
input - 0 (zero).

APOGEE [synctopctime, loadfpga, input, synchmode, externalref,1ppssynch, leapyear]

Refer to the Apogee manual for more details.

synctopctime- gets the current PC time and programs the Apogee board with it.
loadfpga - loads the FPGA in path\filename.
input -
synchmode - always sync to IRIG or not listen to IRIG.
externalref - whether 10 MHz external reference is used.
1ppssynch - syncs to 1pps.
leapyear - tells whether leap year or not.

DISABLE [module#, all, display, ping, debug, multicaststatus]

This command disables a particular stream (module), all streams, or the display, depending on the variable that follows the command.

module# - input a module number with the *disable* command to deactivate a particular module.
all - this variable deactivates all modules currently loaded.
display - this variable turns the Server display off.
ping - this variable disables the network assurance ping that is sent by any open TCP receiver socket within the Avtec PTP for Windows. The network assurance ping is used to make sure that the TCP connection is still "alive". This is enabled by default.
debug - this variable disables the logging of debug information to a file.
multicaststatus – disables multicast status messages. Off by default. Also see command *DISABLEMULTICASTSTATUS.*

DISABLEMULTICASTSTATUS

Disables multicast status messages. Same as *DISABLE multicaststatus*.

ENABLE [module#, all, display, ping, debug, multicaststatus]

This command enables a particular stream (module), all streams, or the display, depending on the variable that follows the command.

module# - input a module number with the *enable* command to activate a particular module.
all - this variable activates all modules currently loaded.
display - this variable turns the Server display on (the display defaults to on whenever the black-box is first loaded).
ping - this variable enables the network assurance ping that is sent by any open TCP receiver socket within the Avtec PTP for Windows. The network assurance ping is used to make sure that the TCP connection is still "alive". This is enabled by default.
debug - this variable enables the logging of debug information to a file. Debug information is stored in the file *%SystemRoot%\ptpdebug.txt*, where *%SystemRoot%* is the system's root directory (usually *C:*).
multicaststatus – enables multicast status messages. Also see command *ENABLEMULTICASTSTATUS.*

ENABLEMULTICASTSTATUS

Enables multicast status messages. Same as *ENABLE multicaststatus.*

EXIT

This command closes the Server program.

LOADAPOGEEFPGA [path\filename]

Loads the Apogee FPGA. The complete path and filename must be given.

LOADDESKTOP [path\filename]

This command opens the *Load Desktop* window box, or, when followed by a path and filename, opens the specified desktop. The default file extension is *DTP*.

NEWDESKTOP

This command removes all currently loaded modules from the desktop and returns an empty Server desktop to the user.

Q

This command closes the Server program.

QUIT

This command closes the Server program.

READSCRIPTFILE [path\filename]

This command is used to execute a Avtec PTP for Windows Script File that has been written. Simply follow the command with the path and filename of the script file (default extension is *PTP*), and the file will be executed.

REMOVEEVENT [source] [destination] 0 0

This command allows an event connection to be removed from between two modules.

source - event source module number.
destination - event destination module number.
0 - future options, currently must be set to 0 (zero).

REMOVEMODULE [module#]

Removes the specified module from the Avtec PTP for Windows desktop.

REMOVEOUTPUT [source] [destination] [port#] [input]

This command allows a data flow command to be removed from between two modules.

source - data stream source module number.
destination - data stream destination module number.
port# - port number of source module to send data from. Valid values are 1 - 32.
input - future option, currently must be set to 0 (zero).

REWINDDEBUGFILE

Empties the file and starts at the beginning again.

SAVEDESKTOP [path\filename]

This command opens the *Save Desktop* window box, or, when followed by a path and filename, directly saves the current desktop. The default file extension is *DTP*.

SECUREBUFFERMODE [enable, disable]

This command enables or disables *Secure Buffer Mode* operation of the Avtec PTP for Windows. When data is passed from one module to another, what is actually passed is a "pointer" to the data. If the same data is passed to multiple modules, all modules that are to receive the data will get the same pointer. This can be a problem if any one of these modules is going to alter the data in any way. With *Secure Buffer Mode* enabled, when one data stream is passed to multiple modules, a copy of the data is made for each module. In this case, if one module changes the data, it will not affect the data being used by the other modules.

enable - enables Secure Buffer Mode.
disable - disables Secure Buffer Mode.

SETACTIVITYDELAY [x]

This command configures the time delay, in milliseconds, between activity status updates of a module. The activity status shows whether a module is currently performing a task. Activity status is displayed in the Server window, in the rows marked *Rx Status* and *Tx Status*, using a 0 (inactive) or 1(active). The first column represents module 1, the second module 2, and so on, up to 32 modules.

SETAPPLICATIONBUFFERS [x]

Sets the application buffer count to *x* values.

SETCONTROLPORT [port#]

This command sets the control port for the Server, which is the socket number assigned for controlling the system. Valid values are 0 through 65536.

SETDEBUGMODE [x]

This command sets the level of detail used when the Server is recording debug information in a debug log file. Valid values for the parameter *x* are 0 – 255. The higher the number, the more detailed the information that is logged. Also see commands *ENABLE debug* and *DISABLE debug*.

SETDEFAULTPATH [path]

Sets the default path to the working directory in *path* and appends \ptp_user\… to the path. Assumes the working directory in *path* has already been created. Fully qualified path cannot end in \. For example, the command **SETDEFAULTPATH c:\frog** will set the path to be *c:\frog\ptp_user\…*

SETFREQUENCY [module#] [frequency]

This command sets the transmitter frequency, in Hertz (Hz), for the Serial Output module only. The command line must include the module number of the Serial Output module and the desired frequency.

SETMODULETTL [module#] [x]

This command sets the number of hops (*x*) a status message from a particular module takes before it is killed.

SETMULTICASTADDRESS [address]

This command sets the Class D multicast IP address to which Avtec PTP for Windows status messages are sent.

SETMULTICASTNIC [xxx.xxx.xxx.xxx]

This command allows the Server status messages to be routed to a user specified Network Interface Card (NIC) where [xxx.xxx.xxx.xxx] is the NIC's address.

SETMULTICASTPORT [port#]

This command sets the multicast port address on the local machine to the port number indicated. The multicast port address sets which port Avtec PTP for Windows status messages are sent out on. Valid values are 0 through 65536.

SETMULTICASTTTL [x]

This command sets the number of hops (*x*) a status message takes before it is killed. This is a global command for all modules.

SETNETWORKBUFFERCOUNT [x]

Sets the network buffer count to the value *x*. Valid values for *x* are 16 – 256.

SETOSVAL [x]

Determines whether the OS is Windows NT or not. 0 = WinNT, 1 = not WinNT.

SETPROCESSPRIORITY [x]

Sets the application priority. The default is 256. Set to a lower number for debug. The recommended value is 32, which is normal process priority.

SETSTATIONID [userid]

This command configures the Server's User ID number. The value, *userid*, is a 32-bit number.

SETSTATUSDELAY [x]

This command configures the time delay, in milliseconds, of status updates between modules. For example, module 1's status is updated, then the Server waits *x* ms before updating module 2's status, and so on through all the modules. Once the last module's status display is updated, the loop starts again at module 1. Therefore, if there are 7

modules loaded on the desktop with a status delay of *x* ms, then each module's display will be updated every *7x* ms.

SYNCSTATIONTIMETHENFREERUN

Assumes a time board is connected. Synchronizes time board's time to station time (IRIG). If fails, sets time board's time back to PC time. After the test is finished, the syncmode is put in free run mode.

USECRITICALPOSTBUFFER [x]

Once a thread has begun, it is guaranteed to run to completion unless a higher thread intervenes. Sleep 0 will give up to a thread of equal or higher value. This command allows you to bypass the Sleep mode and put in in-line function call again. If the value *x* is not zero, use critical post buffer is set to *TRUE*.

0 - load balance the CPU, give up time between modules.
1 - make in-line function calls.

VIEW [module#, que, stats, normal, enable, disable]

The *view* command adjusts the display of the Server window.

module# - using the *view* command with a module number displays the status for that particular module.
que - shows the que (buffer) usage for the current module.
stats - shows the statistics normally displayed for the current module.
normal - same as *stats*.
enable - enables the Server display (actively shows module statistics).
disable - disables the Server display (shows no module/stream/status information).

WAIT [x]

Used when writing Script Files, this command inserts an "x" ms pause between the execution of commands.

ZERO [stream, all]

This command zeros the count displays for a particular stream (module), or all streams, depending on the variable that follows the command.

stream - input a module number with the *zero* command to set the count displays to zero for that particular module.
all - all module counts are set to zero.

Appendix B:

IPDU HEADER SUMMARY

Table B-1 below gives the complete IPDU header structure. The three "Use for" columns indicate which header entries are meaningful for each of the three IPDU data types. The header is fixed at 32 bytes with unused fields set to zero.

Table B-1: IPDU Header Structure

Item No.	Field Name	Format & Size	Value	Use for Tlm	Use for Cmd	Use for Trk
1	IPDU synchronization	unsigned integer (4 bytes)	74C2472C hex	x	x	x
2	IPDU length in bytes (everything from the start of IPDU sync field through the end of the IPDU)	unsigned integer (4 bytes)		x	x	x
3	Data type ID, consisting of:	(4 bytes total)				
3a	IPDU Source	unsigned integer (8 bits)	ID for source of IPDU (see **Table B-2** for values)	x	x	x
3b	IPDU Destination	unsigned integer (8 bits)	ID for destination of IPDU (see **Table B-2** for values)	x	x	x
3c	Message Type	unsigned integer (8 bits)	(see **Table B-3** for values)	x	x	x
3d	Spare	(8 bits)	0			
4	Header version number	unsigned integer (4 bits)	1	x	x	x
5	Data type	unsigned integer (4 bits)	0 (unused for L7)			
6	Message Time (GMT)	NASA PB-5 Code (Option C) (7 bytes)	(See **Table B-4** for values)	x	x	x
7	Ground station physical port ID	unsigned integer (1 byte)	0 (unused for L7)			
8	Source VCDU sequence counter discontinuity	logical (1 bit)	0 = no source VCDU discontinuity 1 = source VCDU discontinuity detected	x		

Item No.	Field Name	Format & Size	Value	Use for Tlm	Use for Cmd	Use for Trk
9	VCDU contains playback data	logical (1 bit)	0 = real-time telemetry 1 = playback telemetry	x		
10	Recovery processing indicator	logical (1 bit)	0 = live from spacecraft 1 = playback from ground station	x		x
11	Test data indicator	logical (1 bit)	0 = operational data 1 = test data	x	x	x
12	CRC failure indicator	logical (1 bit)	0 (unused for L7)			
13	Path SDU source sequence counter discontinuity	logical (1 bit)	0 (unused for L7)			
14	Packet length error	logical (1 bit)	0 (unused for L7)			
15	Packet fill indicator	logical (1 bit)	0 (unused for L7)			
16	Spare	(2 bits)	0			
17	Source VCDU ID, consisting of:	unsigned integer (14 bits total)				
17a	Spacecraft ID	unsigned integer (8 bits)	15 hex for L7	x	x	x
17b	Virtual Channel ID	unsigned integer (6 bits)	All VCDUs for L7 telemetry have a VCID = 0	x		
18	Location of first octet of ground-generated fill data for a path SDU	unsigned integer (2 bytes)	0 (unused for L7)			
19	Spare	(4 bytes)	0			
20	Reed-Solomon error control flag	logical (1 bit)	0 = no errors or errors were corrected 1 = uncorrectable errors	x		
21	Source VCDU header error decode results	unsigned integer (5 bits)	0 (unused for L7)			
22	Source VCDU error decode results	unsigned integer (10 bits)	This field is applicable only if the value of the Reed-Solomon error control flag = 0: 0 = no errors, If > 0, the number of corrected bits within the entire VCDU	x		
	Total Header Length	32 bytes				

Table B-2: L7 IPDU Source and Destination Codes

Source/Destination Name	Corresponding Integer Code (8 Bits) (Hex)	Corresponding 3 Character Ascii Code
Landsat-7 MOC	01	L7M
LGS, Sioux Falls, SD	02	SFS
AGS	03	ASK
SGS	04	SPZ
EOS White Sands Communications	05	WSC
Weilheim, Germany	06	WIL
Valley Forge, PA	07	VLF
Vandenburg AFB, CA	08	VAF
WPS	60	WFF

Table B-3: L7 IPDU Data Types

Message Type	Corresponding Integer Code (8 bits) (hex)
Narrowband real-time telemetry	01
Narrowband spacecraft recorder telemetry	02
Command data message	03
Command echo message	04
One-way tracking data	05
Two-way tracking data	06

Table B-4: NASA PB-5 Time Code Format (Option B)

Item No.	Field Name	Format & Size	Value
1	Flag bit	logical (1 bit)	1
2	Truncated Julian Day	unsigned integer (14 bits)	Truncate the most significant decimal digits, retaining only the four least significant ranging from 0 to 9999. The current Julian day epoch begins at midnight 1995, October 9, 10. October 10, 1995 is day 0.
3	Seconds of Day	unsigned integer (17 bits)	range = 0 to 86399
4	Milliseconds of Second	unsigned integer (10 bits)	range = 0 to 999
5	Microseconds of a Millisecond	unsigned integer (10 bits)	range = 0 to 999
6	Spare	(4 bits)	
	Total Length	7 bytes	

Table B-5: EDOS Service Header

Item No.	Name	Format & Size	Data Characteristics
1	ESH Version number	Unsigned Integer 4 Bits	Range -> 0- 15. (0=intitial version, and incremented by 1 for each ESH modification thereafter)
2	SDU Type	Unsigned Integer 4 Bits	Value -> 0=VCDU, 1=CCSDS Packet (Path SDU), and 2=CLCW. Note: AM-1 provides VCDUs for the spacecraft AM-1 Trash Buffer
3	Date and Time Annotation of return link SDU receipt at EDOS.	NASA PB-5 Code Format 7 Bytes	Range -> Refer to **Table B-4** for the NASA PB-5 Code Format (containing GMT).
4	EDOS Physical Port Identification	Unsigned Integer 1 Byte	Range -> 0-63
5	Source VCDU Sequence Counter Discontinuity	Logical Bit 1 Bit	Values -> 0=false (No source VCDU discontinuity) and 1=true (source VCDU discontinuity)
6	VCDU contains Playback data	Logical Bit 1 Bit	Values -> 0 = False (nonplayback data) and 1 = True (playback data) (from the on-board spacecraft recorder)
7	Recovery Processing Indicator	Logical bit 1 Bit	Values -> 0 = false (live) and 1= true (data capture playback) (from EDOS's data capture recovery processing)
8	Test data indicator	Logical Bit 1 Bit	Values -> 0 = false (live-operational data) and 1 = true (EDOS test data)
9	Cyclic Redundancy Check (CRC) failure indicator	Logical Bit 1 Bit	Field is present, but it is not applicable for AM-1. AM-1 is a Grade 2 service using Reed-Solomon. Value = 0 (FALSE) for AM-1
10	Path SDU Source Sequence Counter Discontinuity	Logical Bit 1 Bit	Values -> 0 = false (no discontinuity); 1 = true (packet discontinuity)
11	Packet Length Error	Logical Bit 1 Bit	Value -> 0=no detected length error; 1=detected length error.
12	Packet Fill Indicator	Logical Bit 1 Bit	Value -> 0=packet doesn't contain fill data; 1=packet contains fill data.
13	Fill/Spare, reserved for future use.	Unsigned Integer 2 Bits	Value -> zero (0)
14	Source VCDU-ID	Unsigned Integer 14 Bits	Spacecraft ID and VCID (SCID-8 Bits and followed by VCID-6 Bits (Extracted for VCDU))
15	Location of first octet of EDOS generated fill data for a Path SDU	Unsigned Integer 2 Bytes	Range -> 0 - 65535
16	Fill/Spare, reserved for future use	Unsigned Integer 4 Bytes	Value -> zero (0).

Item No.	Name	Format & Size	Data Characteristics
17	Reed-Solomon Error Control Flag	Logical Bit 1 Bit	Values -> 0=correctable or no errors; 1=Failed/uncorrectable Flag indicates whether the VCDU failed or passed R-S decoding.
18	Source VCDU Header Error Decode Results	Unsigned Integer 5 Bits	Values -> 0=error free. Field contains the number of corrected symbols within the VCDU header.
19	Source VCDU error decode results	Unsigned Integer 10 Bits	Values -> 0=error free Field contains the number of corrected symbols from the entire VCDU.
The EDOS Service Header has a total of twenty bytes			

Table B-6: NASCOM RTP Header

Item No.	Name	Size	Data Characteristics
1	version (V)	2 bits	This field identifies the version of RTP.
2	Padding (P)	1 bit	If the padding bit is set, the packet contains one or more additional padding octets at the end which are not part of the payload.
3	Extension (X)	1 bit	If the extension bit is set, the fixed header is followed by exactly one header extension, with a format as defined in RFC 1889, Section 5.3.1.
4	CSRC count (CC)	4 bits	The CSRC count contains the number of CSRC indentifies that follow the fixed header.
5	Marker (M)	1 bit	The interpretation of the marker is defined by a profile. A profile may define additional marker bits or specify that there is no marker by changing the number of bits in the payload type field. (see RFC 1889, Section 5.3)
6	Payload type (PT)	7 bits	This field identifies the format of the RTP payload and determines it interpretation by the application. A profile specifies a default static mapping of payload type codes to payload formats.
7	Sequence number	16 bits	The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number is random.
8	Timestamp	32 bits	The timestamp reflects the sampling instant of the first octet in the RTP data packet. The resolution of the clock must be sufficient for the desired synchronization accuracy and for measuring packet arrival jitter. The clock frequency is dependent on the format of data carried as payload and is specified statically in the profile or payload.
9	SSRC	32 bits	The SSRC field identifies the synchronization source. This identifier is chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier.

Table B-7: LEO-T Telemetry Frame Delivery Header (TFDH)

Word	Bit(s)	Description
1	1-2	Version number of telemetry header
1	3-16	Length of frame in bytes (including header)
2	1	Annotation Flags: Reed-Solomon (RS) decoding enabled flag
2	2	RS decoding error flag
2	3	Frame CRC enabled flag
2	4	Frame CRC pass/fail flag
2	5	Master channel sequence (MCS) checking enabled flag
2	6	MCS number error
2	7-8	Data inversion flags
2	9-10	Fame sync make flags
2	11	Data forward/reverse flag
2	12-16	Data Class
3	1	Earth received time of data (PB-5 format): flag bit
3	2-15	Earth received time of data (PB-5 format): truncated Julian Day
3	16	Earth received time of data (PB-5 format): seconds of day
4	1-16	Earth received time of data (PB-5 format): seconds of day
5	1-10	Earth received rime of data (PB-5 format): milliseconds of a second
5	11-16	Fill / spare

Note: 'word' refers to 16 bits, an 'octet' or 'byte' refers to 8 bits.

The fields in the LEO-T telemetry frame delivery unit are defined as follow. Bit 1 is the most significant bit and is transmitted first.

Word 1

Bits 1-2: Version number for header format is set to "01" for frame data.
Bits 3-16: Length of frame in bytes, including frame delivery header.

Word 2

Bit 1: Reed-Solomon decoding:
"0" disabled
"1" enabled
Bit 2: RS decoding error:
"0" Frame was corrected
"1" Frame could not be corrected
Bit 3: Frame CRC checking:
"0" disabled
"1" enabled
Bit 4: Frame CRC error:

"0" pass
"1" fail

Bit 5: Master Channel Sequence (MCS) checking:
"0" disabled
"1" enabled

Bit 6: MCS number error:
"0" number increased monotonically
"1" number increased by 2 or more

Bit 7, 8: Data inversion flags:
"00" data true
"01" data inverted
"11" data inverted and corrected

Bit 9, 10: Frame sync mode flags:
"0" search frame
"01" check frame
"10" lock frame
"11" flywheel frame

Bit 11: Data forward/reverse flag:
"0" forward
"1" reverse

Bit 12-16: Data Class:
1 = CCSDS Frame
2 = CCSDS Packet
3 = TDM Frame
4 = Stripped TDM Frame

Words 3-5

Earth received time as defined below. This is the UTC time when the telemetry frame is received by the telemetry processor. The accuracy of the time is 1 millisecond.

Word 3

Bit 1: PB-5 flag bit, set to "0"

Bits 2-15: Truncated Julian day (14 bits). Truncate the most significant decimal digits, retaining only the four least significant decimal digits ranging from 0000 to 9999. The current Julian day epoch begins on October 10, 1995, and continues for a period of 27 years. Value is binary unsigned integer.

Bit 16: Seconds of day (17 bits), most significant bit.

Word 4

Bits 1-16: Seconds of day (17 bits), 16 least significant. Value is binary unsigned integer, ranging from 0 to 86,399.

Word 5

Bits 1-10: Milliseconds. Value is binary unsigned integer from 9 to 999.

Bits 11-16:Fill/spares, set to zero's.

Table B-8: LEO-T Command Delivery Header (CDH)

BYTE(S)	Description
1	Message Type
2	Fill/Spare - Reserved for Future Use
3	Source Identification
4	Destination Identification
5	Fill/Spare - Reserved for Future Use
6-12	Message generation time - NASA PB-5 Time code format
13-14	Spacecraft Identification
15-16	Message Sequence number
17-18	EDOS software version number
19-20	Message Length
21-24	Fill/Spare - Reserved for future use
Command data field	

Note: 'word' refers to 16 bits, an 'octet' or 'byte' refers to 8 bits.

The fields in the TRACE command delivery header are defined as follows. Bytes are transmitted MSB first. Fields that are denoted with '*' are command validation parameters and shall be included in the configuration database. Remaining fields (except for Message Length) are not used by the TRACE and are for information purposes only.

Byte 1: * Message type is 03 hex (Command data block)
Byte 2: Fill/spares, set to zero's.
Byte 3: * Source Identification
Byte 4: Destination Identification
Byte 5: Fill/spares, set to zero's

Bytes 6-12: Earth received time as defined below. This is the UTC time when the command ground transfer unit is generated.

Bytes 6, 7
- Bit 1: PB-5 flag bit, set to "0"
- Bits 2-15: Truncated Julian day (14 bits). Truncate the most significant decimal digits, retaining only the four least significant decimal digits ranging from 0000 to 9999. The current Julian day epoch begins October 10, 1995, and continues for a period of 27 years. Value is binary unsigned integer.
- Bit 16: Seconds of day (17 bits), most significant bit.

Bytes 8, 9

Bits 1-16: Seconds of day 17 (bits), 16 least significant. Value is binary unsigned integer, ranging from 0 to 86,399.

Bytes 10, 11

Bits 1-10: Millisecond (10 bits). Value is binary unsigned integer from 0 to 999.

Bits 11-16: Microseconds of milliseconds (10 bits), 6 most significant. Value is binary unsigned integer from 0-999.

Bytes 12

Bits 1-4: microseconds of milliseconds, 4 least significant

Bits 5-8: Fill, set to zero's

Bytes 13,14: * Spacecraft ID

Bytes 15,16: Message sequence number

Bytes 17,18: EDOS software version number, set to TBD

Bytes 19,20: Message Length. Number of bytes in the message, including the LEO-T command delivery header and the command data. Range is 24 to 8192.

Bytes 21-24: Fill/spares, set to zero's

Appendix C:

ACRONYMS AND ABBREVIATIONS

APID	Application ID
BERT	Bit Error Rate Tester
BIΦL	Bi-phase Level
BIΦM	Bi-phase Mark
BIΦS	Bi-phase Space
BIT SYNC	Bit Synchronization
BPSK	Binary Phase Skift Keying
CCSDS	Consultative Committee for Space Data Systems
CLK	Clock
CLTU	Command Link Transmission Unit
COM	Communications
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DLL	Dynamic Link Library
DQM	Data Quality Monitor
Hex	Hexadecimal
IDE	Integrated Drive Electronics
I/O	Input/Output
IP	Internet Protocol
IPDU	Inter-project Data Unit
IRIG	Inter-range Instrumentation Group
IRQ	Interrupt Request
ISA	Industry Standard Architecture

Acronyms and Abbreviations - *(continued)*

LAN	Local Area Network
LSB	Least Significant Bit
MB	Megabyte
Mbps	Megabits Per Second
MHz	Megahertz
MSB	Most Significant Bit
msec	millisecond
NRZL	Non-return to Zero Level
NRZM	Non-return to Zero Mark
NRZS	Non-return to Zero Space
PB	Parallel Binary *(can be numbered 1 through 5)*
PCI	Peripheral Component Interface
PCM	Pulse Code Modulation
PSK	Phase Shift Key
PTP	Programmable Telemetry Processor
ROM	Read-only Memory
RS	Reed-Solomon
Sync	Synchronizer
SCSI	Small Computer Systems Interface
TCP	Transmission Control Protocol
TDM	Time-division Multiplex
TTL	Time to Live
UDP	User Datagram Protocol
Uncon.	Unconnected
VCDU	Virtual Channel Data Unit
VCID	Virtual Channel ID
VCP	Virtual Channel Processor
WAN	Wide Area Network